VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ

FACULTY OF MECHANICAL ENGINEERING

ÚSTAV MECHANIKY TĚLES, MECHATRONIKY A BIOMECHANIKY

INSTITUTE OF SOLID MECHANICS, MECHATRONICS AND BIOMECHANICS

SESTAVENÍ A VYTVOŘENÍ ÚLOH PRO INTERAKTIVNÍ ROBOTICKOU HLAVU

ASSEMBLING AND CREATING TASKS FOR AN INTERACTIVE ROBOTIC HEAD

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Michal Szabó**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Michal Bastl**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadaní bakalářské práce

Ústav:	Ústav mechaniky těles, mechatroniky a biomechaniky
Student:	**Michal Szabó**
Studijní program:	Aplikované vědy v inženýrství
Studijní obor:	Mechatronika
Vedoucí práce:	**Ing. Michal Bastl**
Akademický rok:	2020/21

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Sestavení a vytvoření úloh pro interaktivní robotickou hlavu

Stručná charakteristika problematiky úkolu:

Práce se zabývá rozšířením a vývojem uživatelského rozhraní pro model robotické hlavy. Návrh hlavy je hotov a v rámci práce bude jen převzat a sestaven dle dokumentace. Sestava robohlavy je osazena senzorem RealSense, kterým je možné získávat obraz a hloubkovou mapu. Úkolem práce bude implementovat úlohy pro interakci s uživatelem na základě zpracování obrazových dat a pomocí syntetizované řeči.

Cíle bakalářské práce:

1. Vytvořte rešerši k problematice interaktivních robotických hlav. Dále rešerší o problematice dostupných nástrojů k rozpoznaní objektů v obraze a k rozpoznání mluvené řeči (angličtina).
2. Sestavte robohlavu dle dostupné dokumentace. Funkční návrh robohlavy je již hotov. Sestavu oživte a ověřte funkčnost.
3. Nainstalujte a zprovozněte potřebné nástroje (Python + potřebné knihovny).
4. Vytvořte interaktivní hry s uživatelem. Rozpoznávání předmětů. Jednoduchá slovní komunikace apod.

Seznam doporučené literatury:

VALÁŠEK, Michael. Mechatronika. Praha: České vysoké učení technické, 1995. ISBN 80-01-01276-X.

CORKE, Peter. Robotics, vision and control: fundamental algorithms in MATLAB. 2nd ed. Berlin: Springer, 2013. Springer tracts in advanced robotics. ISBN 978-3-642-20143-1.

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2020/21

V Brně, dne

L. S.

prof. Ing. Jindřich Petruška, CSc. ředitel ústavu	doc. Ing. Jaroslav Katolický, Ph.D. děkan fakulty

Abstrakt

Táto bakalárska práca sa zaoberá vytvorením modelu interaktívnej robotickej hlavy. Samotná práca je rozdelená do dvoch častí, teoretickej a praktickej. Teoretická časť je venovaná prehľadu typov robotických hláv, stručnému opisu dostupných nástrojov na rozpoznávanie objektov v obraze a nástrojov na rozpoznávanie hovorenej reči v reálnom čase. Praktická časť je zameraná na nástroje použité pri programovaní daného modelu, na použitú elektroniku a výsledný model robotickej hlavy. Napokon sú v nej opísané naprogramované funkcie umožňujúce rôzne spôsoby interakcie s človekom. Súčasťou práce sú skripty funkcií naprogramované v jazyku Python.

Summary

This bachelor's thesis deals with the creation of the model of an interactive robotic head. The work itself is divided into two parts, theoretical and practical. The theoretical part is devoted to an overview of the types of robotic heads, a brief description of the available tools for recognizing objects in the image and tools for recognizing spoken speech in real time. The practical part is focused on the tools used in programming of this model, the electronics used and the resulting model of the robotic head. Finally, there are described programmed functions enabling various ways of the interaction with humans. The work includes function scripts programmed in Python.

Kľúčové slová

robohlava, robotická hlava, RealSense D435, Arduino, YOLO, syntetizovaná reč, Google Speech-To-Text, Mechatronika

Keywords

robo-head, robotic head, RealSense D435, Arduino, YOLO, synthesized speech, Google Speech-To-Text, Mechatronics

SZABÓ, M. *Sestavení a vytvoření úloh pro interaktivní robotickou hlavu.* Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2021. 38 s. Vedoucí Ing. Michal Bastl.

Čestne prehlasujem, že táto bakalárska práca na tému *„Sestavení a vytvoření úloh pro interaktivní robotickou hlavu“* bola mnou samostatne vypracovaná pod vedením Ing. Michala Bastla a že som uviedol všetky použité pramene a literatúru.

Michal Szabó

Chcel by som sa poďakovať pánovi Ing. Michalovi Bastlovi za cenné rady a vedenie tejto bakalárskej práce.

Ďalej by som sa chcel poďakovať Bc. Artyomovi Voroninovi, ktorý bol autorom kódu predošlej verzie robohlavy, za vysvetlenie jednotlivých funkcií a pomocné rady pri programovaní práce. Vďaka patrí aj Ing. Martinovi Appelovi za pomoc pri modelovaní a následnom vytlačení dielov na 3D tlačiarni.

Michal Szabó

Obsah

Úvod

Technológie sa vyvíjajú veľmi rýchlo a je len otázkou času, kedy sa pomocou algoritmov umelej inteligencie a strojového učenia zdokonalia na takú mieru, že predbehnú ľudskú inteligenciu. Je veľmi pravdepodobné, že ak sa dostaneme do tohto bodu, nebudeme schopný pochopiť všetko, čo počítače dokážu. V mnohých sférach dokázala umelá inteligencia súčasnosti prekonať schopnosti človeka, avšak klasické počítače sa už dostávajú na pokraj svojich možností, preto sa začali vyvíjať iné metódy výpočtovej techniky, akými sú napríklad kvantové počítače, ktoré nám umožnia riešenie komplexnejších problémov a rýchlejšie spracovanie dát. S takouto technikou bude možné vytvoriť systémy, ktoré chápu a reagujú na podnety z vonkajšieho prostredia oveľa rýchlejšie ako ľudia.

Táto bakalárska práca sa zaoberá vývojom modelu interaktívnej robotickej hlavy. Práca je rozdelená do dvoch základných celkov, a to do teoretickej a praktickej časti.

Prvá kapitola teoretickej časti prináša roztriedenie interaktívnych robotických hláv a približuje problematiku návrhu ich hardwaru a softwaru. Ďalšia kapitola je venovaná stručnému zhrnutiu dostupných nástrojov na rozpoznávanie objektov v obraze, ich základnej podstate a názornému porovnaniu. Následne sumarizuje dostupné nástroje na rozpoznávanie hovorenej reči v reálnom čase.

Praktická časť je venovaná zostaveniu modelu robotickej hlavy a pomocou nástrojov uvedených v predošlej časti naprogramovaniu jednoduchých interaktívnych hier s užívateľom v jazyku Python. V prvej kapitole sú zhrnuté všetky použité knižnice a nástroje využité pri jej programovaní. Druhá kapitola prináša opis návrhu a zostavenia provizórnej konštrukcie robotickej hlavy. V poslednej kapitole sú vysvetlené jednotlivé funkcie pomocou zdokumentovaných obrázkov.

1 Teoretická časť

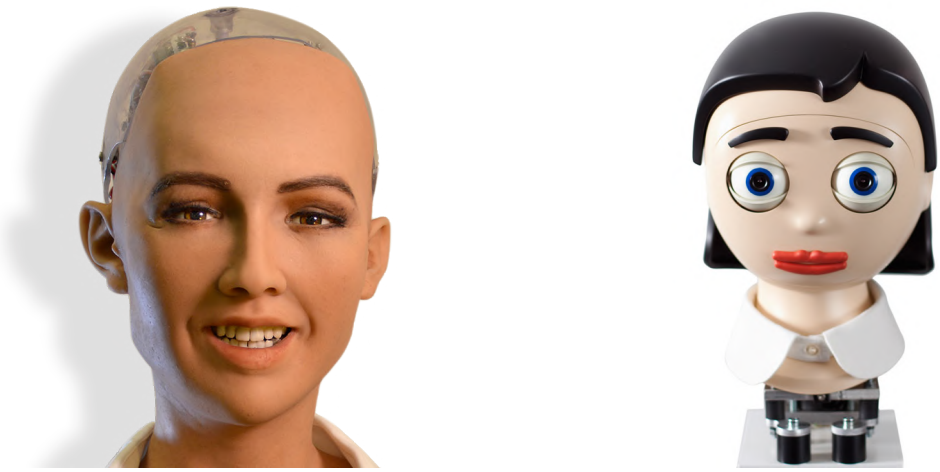
V tejto časti sú uvedené základné poznatky a rozdelenie interaktívnych robotických hláv. V stručnosti sú popísané aj základné myšlienkové postupy na návrh dizajnu a rôznych interaktívnych funkcií pre model robotickej hlavy. Ďalej sú tu spomenuté a zvlášť vysvetlené dostupné nástroje na rozpoznávanie objektov v obraze a princípy, na ktorých fungujú. V poslednej časti sú zhrnuté dostupné nástroje na rozpoznávanie hovorenej reči, ktoré je možné využiť pri programovaní aplikácií.

1.1 Interaktívne robotické hlavy

Interaktívne robotické hlavy (IRH) možno rozdeliť podľa ich vzhľadu, funkcie a charakteristiky správania sa do troch základných skupín, a to na antropomorfné, zoomorfné a technomorfné [1].

Antropomorfné roboty

Antropomorfné roboty pripomínajú ľudský vzhľad, pohyb a taktiež aj správanie. Príkladom môže byť robot Sophia (obr.1.1a) [2] od spoločnosti Hanson Robotics, ktorý sa v súčasnosti považuje za jeden z najvyspelejších interaktívnych robotov, robotická hlava Flobi (obr.1.1b) [3] vytvorená na Belfieldskej univerzite a iné.



(a) Sophia. (b) Flobi.

Obr. 1.1: Antropomorfné robotické hlavy

Zoomorfné roboty

Zoomorfné roboty slúžia prevažne na zábavu a svojím zovňajškom a správaním pripomínajú živé bytosti. Tieto roboty sú schopné imitovať zvieratá, na základe ktorých boli vytvorené. Príkladom môže byť iCat (obr.1.2a) [4] od firmy Philips, Probo (obr.1.2b) [5] od Vlamskej univerzity v Belgicku a iné.

(a) iCat. (b) Probo.

Obr. 1.2: Zoomorfné robotické hlavy

Technomorfné roboty

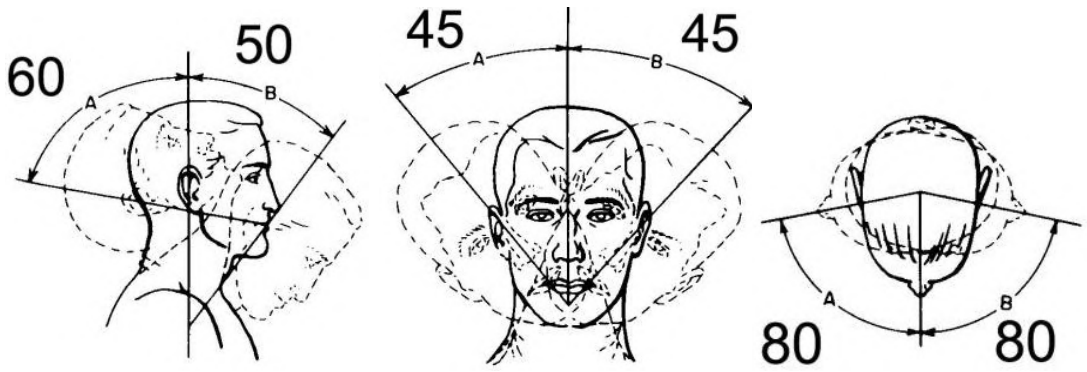
Technomorfné roboty pripomínajú skôr stroje alebo technické konštrukcie. Ich vzhľad a správanie sa nemusí podobať živej bytosti. Takýmto robotom je napríklad robot PR2 (obr.1.3) [6] od spoločnosti Willow Garage.

Obr. 1.3: Technomorfný robot PR2. [6]

1.1.1 Dizajn robotickej hlavy

Návrh IRH spočíva vo vytvorení konštrukcie, ktorá dokáže presne opísať pohyb a vyzerá ako bytosť, ktorú chceme zostrojiť. Príkladom pre objasnenie môže byť ľudská hlava. Pri návrhu ľudskej IRH je potrebné analyzovať všetky typy a možnosti jej pohybu, ako je pohyb krku, hlavy, očí, úst a iných častí tela podľa toho, aký presný by model mal byť.

Pre identifikáciu a získanie informácií o prirodzenom pohybe krku na návrh IRH, je potrebné študovať stavbu tela človeka. Ľudský krk sa skladá zo siedmych stavcov a je to z kinematického hľadiska komplexný mechanický systém. Krčné stavce umožňujú človeku rotačné pohyby hlavy pozdĺž zvislej osy a nakláňacie pohyby hlavy vo všetkých smeroch [9]. Priemerné rozsahy pohybu hlavy sú znázornené na obr.1.4.



(a) Predklon(A), Záklon(B). (b) Náklon vpravo(A), vľavo(B). (c) Rotácia vpravo(A), vľavo(B).

Obr. 1.4: Rozsahy pohybu hlavy človeka [8].

Pohyb hlavy človeka súvisí aj s pohybom očí. Ľudia majú tendenciu otáčať hlavu tam, kam smerujú ich oči. Ľudské oko je taktiež veľmi komplexný mechanický systém schopný vykonávať rôzne typy pohybov [10]. Pre zaujímavosť si uvedieme dva pohyby, ktoré bežne používame:

Sakadické pohyby – veľmi rýchle skokové pohyby očí v rozmedzí niekoľko stoviek metrov za sekundu. K sakadickým pohybom oka dochádza, keď oči nie sú schopné zamerať predmet alebo keď človek hľadá mimo zorného poľa.
Hladké sledovacie pohyby – ide o pohyby, keď oči sledujú objekt. Počas hladkého sledovania sa oči snažia vyrovnať rýchlosti pohybu sledovaného objektu, zvyčajne pri relatívne pomalej rýchlosti.

Rozsah pohybu očí v horizontálnom smere je približne 45° vpravo a vľavo. Vo vertikálnom smere sa oko môže nakloniť smerom nadol o 47° a nahor o 28° [11].

Ďalším dôležitým prvkom IRH je schopnosť napodobniť mimiku tváre. Na ľudskej tvári je 46 mimických svalov. Výrazy tváre sa vytvárajú vďaka pohybu pokožky týmito svalmi. Nie je však praktické napodobniť všetky tieto svaly na robotickej hlave, a to hlavne z dôvodu obmedzenia veľkosti akčných členov a mechanizmov. Preto sa veľa prístupov pokúsilo nakonfigurovať tieto obmedzené stupne voľnosti (DoF - Degrees of Freedom) na dosiahnutie expresívnej robotickej hlavy.

Rozlišujú sa dva štýly výrazov tváre IRH: ľudské výrazy a výrazy, ktoré sú symbolické. Robotická hlava s ľudskými výrazmi tváre má na svojom povrchu pokožku a vytvára mimiku jej deformáciou. Mnoho prístupov sa pokúsilo deformovať pokožku čo najprirodzenejšie. Na druhej strane má robotická hlava so symbolickými výrazmi tváre tuhý povrch a časti tváre na tomto povrchu sú posúvané tak, aby zobrazovali výrazy. V tejto súvislosti sa veľa vedcov pokúsilo dosiahnuť jednoduché a výrazné mechanizmy [12].

Na rozhýbanie týchto pohyblivých častí robotickej hlavy sa naprogramuje vhodný software, ktorý bude optimalizovaný tak, aby čo najreálnejšie imitoval hlavu človeka.

1.1.2 Interakcia robotických hláv

Podľa možností môže IRH reagovať na vonkajšie podnety pomocou základných zmyslov živých bytostí, ktorými sú: zrak, sluch, hmat, čuch a chuť. Všetkým týmto zmyslom musí odpovedať vhodný hardware alebo kombinácia hardwarov na zachytenie danej informácie, ktorú IRH môže spracovať.

Zrak môže byť riešený pomocou bežnej kamery alebo hĺbkovej kamery, ktorá robohlave umožní vypočítať si vzdialenosť od nejakého predmetu. Obraz z kamery sa ďalej spracováva pomocou neurónových sietí a to umožní robohlave rozpoznať objekty v zornom poli kamery. Vhodné nástroje na rozpoznávanie objektov v obraze sú uvedené v sekcii 1.2.

Sluch IRH môže byť riešený využitím mikrofónov. Mikrofón konvertuje akustickú energiu (zvuková vlna) do energie elektrickej (zvukový signál), ktorý IRH dokáže spracovať a pomocou nástroja na rozpoznanie reči rozoznať jednotlivé slová. Tieto nástroje sú uvedené v sekcii 1.3.

Pre vnímanie hmatu, resp. dotyku sa použijú jednoduché tlakové senzory, pomocou ktorých vie IRH zistiť, či sa jej niekto dotýka, a akou silou.

Čuch a chuť sú pomerne zložité zmysly ktoré fungujú na podobnom princípe. Človek má v nose a na jazyku špeciálne receptory, ktoré dokážu rozlíšiť jednotlivé molekuly a následne poslať signál špecifickým častiam mozgu. Implementácia týchto zmyslov do IRH je možná pomocou elektrického nosa (e-nose) a elektrického jazyka (e-tongue) [13].

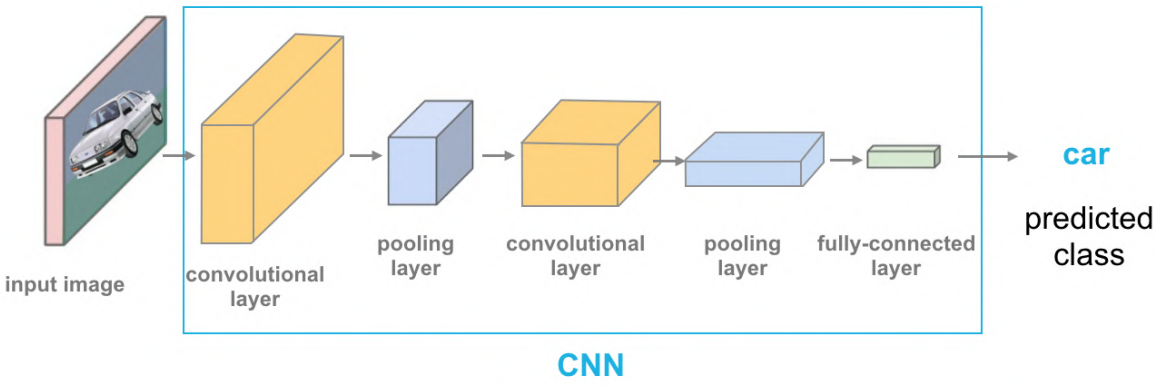
Všetky tieto zmysly vyžadujú rozpoznávanie určitých vzorov (pattern recognition) pre pochopenie informácie z vonkajšieho prostredia. Aj keď IRH pochopí tieto informácie, musí sa na základe nich rozhodnúť, a to je jedna z najťažších úloh pri programovaní takéhoto systému. My dokážeme napísať skript, ktorý to rozhodnutie urobí za ňu, no ak sa má sama učiť a spájať si jednotlivé informácie, ako to robia ľudia, musíme využiť metódy strojového učenia (machine learning) a umelej inteligencie (artificial intelligence), aby sme sa len priblížili komplexnosti ľudskej mysle.

1.2 Nástroje na rozpoznávanie objektov v obraze

Počítačové videnie je interdisciplinárna oblasť, ktorá sa v posledných rokoch (od CNN (1.2)) dosť vyvinula, a to aj z dôvodu oblasti záujmu o samo riadiace vozidlá. Ďalšou neoddeliteľnou súčasťou počítačového videnia je detekcia objektov. Detekcia objektov pomáha pri odhaľovaní pozície, detekcií vozidla, sledovaní atď. Rozdiel medzi algoritmami detekcie objektov a klasifikačnými algoritmami spočíva v tom, že v algoritmoch detekcie sa pokúšame okolo objektu záujmu nakresliť ohraničujúci rámček (bounding box) a umiestniť ho do obrazu. Keďže takýchto objektov môže byť v obraze viac a ich počet nie je vopred známy, tak nastáva problém. To znamená, že dĺžka výstupnej vrstvy konvolučnej siete bude premenlivá. Naivným spôsobom riešenia tohto problému by bolo zobrať zo záberu rôzne oblasti záujmu a pomocou CNN (1.2) klasifikovať prítomnosť objektu v tejto oblasti. Problém tohto prístupu spočíva v tom, že predmetné objekty môžu mať rôzne priestorové umiestnenia v rámci obrázka a rôzne pomery strán. Z tohto dôvodu by sa musel zvoliť obrovský počet oblastí, čo by bolo výpočtovo veľmi náročné. Preto boli vyvinuté nižšie uvedené algoritmy, aby našli tieto výskyty objektov, a to dostatočne rýchlo.

Convolutional Neural Network (CNN)

Convolutional Neural Network v preklade konvolučná neurónová sieť, ktorú používajú tieto detekčné metódy, je algoritmus hĺbkového učenia, ktorý si dokáže zobrať vstupný obrázok a priradiť dôležitosť (naučiteľné váhy a skreslenia - learnable weights and biases) rôznym aspektom/objektom v obraze a vie ich navzájom odlíšiť. Na rozdiel od primitívnych metód, ktorých filtre sú vyrábané ručne, konvolučné neurónové siete sú schopné sa tieto filtre/charakteristiky naučiť. Architektúra systému CNN je obdobná s architektúrou prepojenia neurónov v ľudskom mozgu a bola inšpirovaná organizáciou vizuálneho kortexu človeka [15].

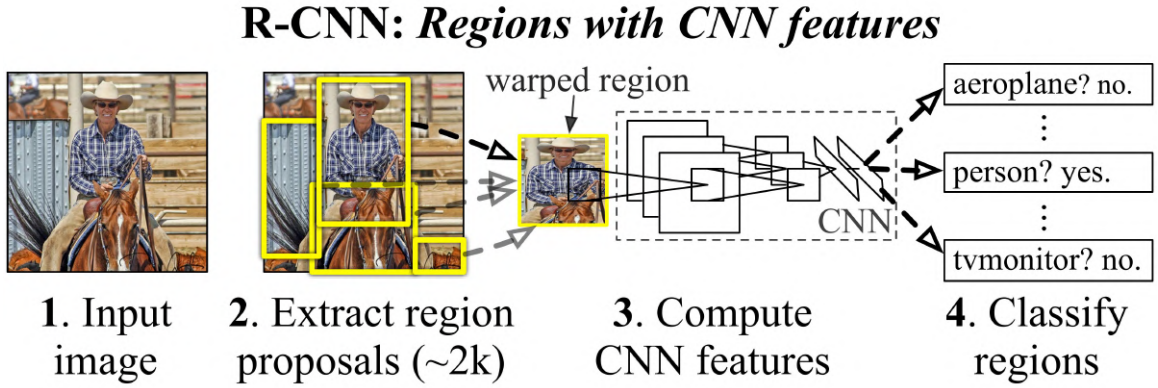


Obr. 1.5: Príklad sekvencie CNN pre klasifikáciu písaných čísel [16].

1.2.1 Region-based Convolutional Neural Network (R-CNN)

Region-based Convolutional Neural Network, teda regionálne založená konvolučná neurónová sieť, rieši problém výberu veľkého počtu oblastí tak, že pomocou selektívneho vyhľadávania extrahuje z obrázka iba 2000 oblastí, ktoré Ross Girshick [17] nazval *region proposals* (navrhované oblasti). Týchto 2000 oblastí sa generuje podľa nasledovného algoritmu:

- generovanie počiatočnej subsegmentácie, čo znamená generovanie veľkého počtu možných oblastí.
- použitie algoritmu vyhľadávania s názvom greedy pre rekurzívne kombinovanie podobných oblastí do väčších celkov.
- vytvorenie finálne navrhovanej oblasti (*region proposals*) z vygenerovaných oblastí.



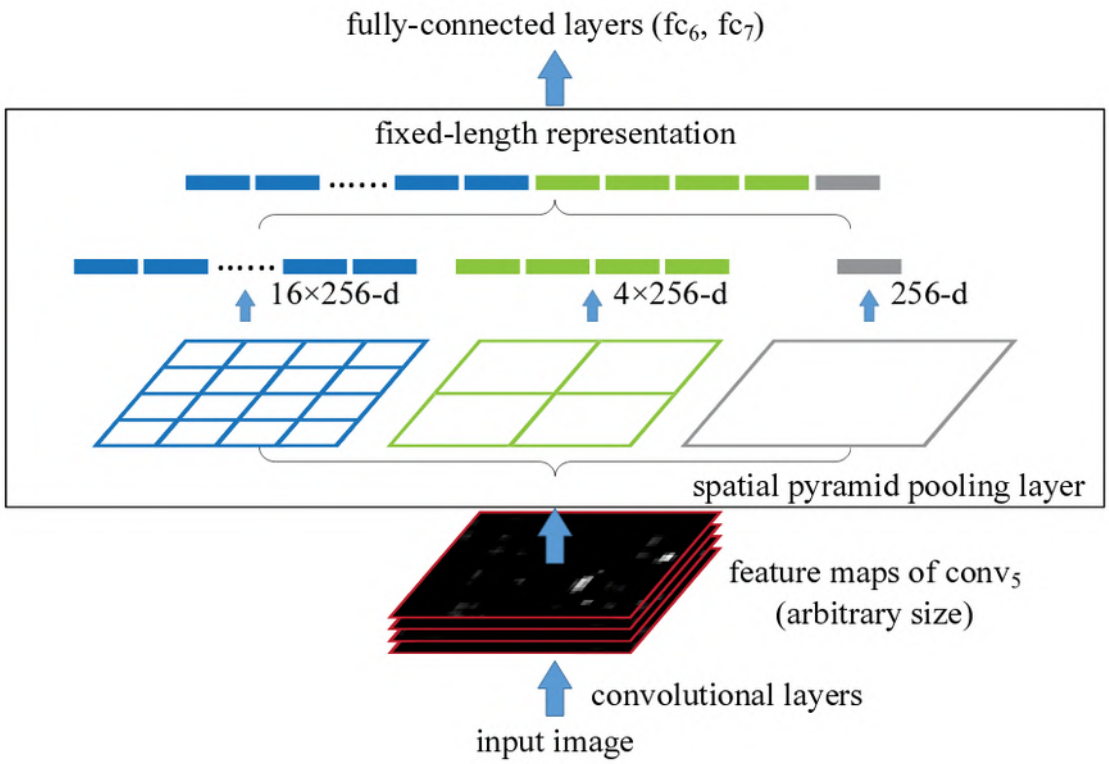
Obr. 1.6: Architektúra R-CNN [17].

Viac informácií o tomto algoritme selektívneho vyhľadávania je dostupných na [18]. Týchto 2000 možných navrhovaných oblastí je spojených do štvorca a odoslaných do konvolučnej neurónovej siete, ktorá vytvorí 4096 dimenzionálny vektor ako výstup. CNN funguje ako extraktor prvkov a výstupná vrstva pozostáva z prvkov extrahovaných z obrázka. Následne sa tieto extrahované prvky privádzajú do podporujúceho vektorového stroja (SVM – Support Vector Machine [19]) na klasifikáciu prítomnosti objektu v danej navrhovanej oblasti. Okrem toho predpovedá aj štyri hodnoty, ktoré sú hodnotami posunu, aby sa zvýšila presnosť ohraničovacieho rámčeka. Napríklad pri určitej navrhovanej oblasti by algoritmus predpovedal prítomnosť osoby, ale polovica tváre tejto osoby by mohla byť odrezaná. Preto hodnoty posunu pomáhajú pri úprave ohraničovacieho rámčeka navrhovanej oblasti.

Problémom R-CNN je, že naučenie neurónovej siete stále trvá obrovské množstvo času, pretože sa musí klasifikovať 2 000 návrhov oblastí pre jeden obrázok. Tým pádom ho nie je možné implementovať v reálnom čase, pretože to trvá asi 49 sekúnd (viď obr.1.10) pre každý skúšobný obrázok. Taktiež je algoritmus selektívneho vyhľadávania pevným algoritmom, preto sa v tomto štádiu nedeje žiadne učenie. To by mohlo viesť k vytvoreniu nesprávnych navrhovaných oblastí.

1.2.2 Spatial Pyramid Pooling (SPP-net)

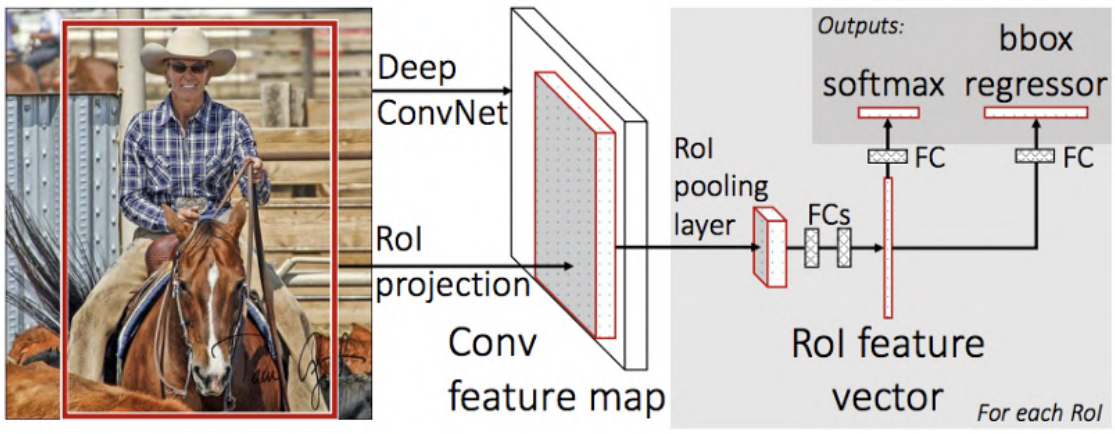
Spatial Pyramid Pooling, v preklade združovanie priestorových pyramíd, je sieťová štruktúra, ktorá môže generovať reprezentáciu pevnej dĺžky bez ohľadu na veľkosť alebo mierku obrázka. O metóde Pyramid Pooling sa hovorí, že je odolná voči deformáciám objektov a SPP-net vylepšuje všetky metódy klasifikácie obrazov založené na CNN (1.2). Pomocou SPP-net môžu vedci vypočítať mapy funkcií z celého obrazu iba raz a potom zhromaždiť prvky v ľubovoľných oblastiach (podobrázkoch), aby vygenerovali reprezentácie pevnej dĺžky na trénovanie detektorov. Táto metóda zabráni opakovanému výpočtu konvolučných funkcií [20].



Obr. 1.7: Sieťová štruktúra s vrstvou SPP [20].

1.2.3 Fast R-CNN

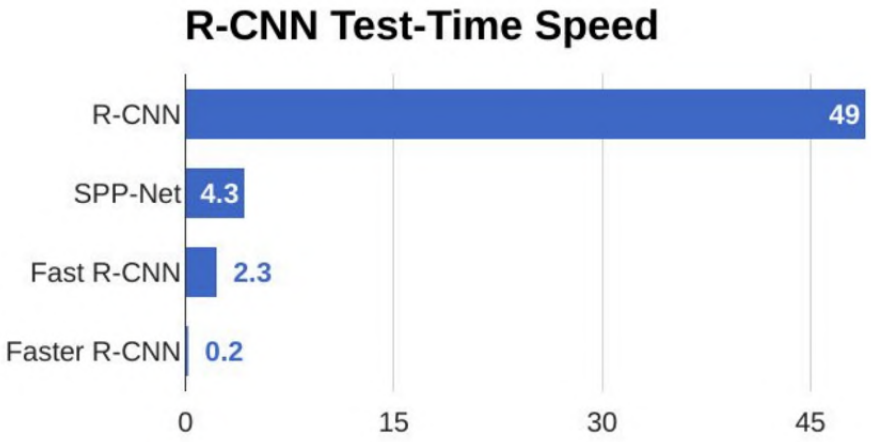
Rovnaký autor predošlého článku (R-CNN 1.2.1) vyriešil niektoré nevýhody R-CNN pre zostavovanie rýchlejšieho algoritmu detekcie objektov a nazval ho Fast R-CNN (Rýchly R-CNN). Prístup je podobný algoritmu R-CNN, ale namiesto toho, aby sa do CNN poslali návrhy oblastí, privádza sa do CNN rovno vstupný obraz, aby sa vytvorila mapa konvolučných funkcií. Z mapy konvolučných funkcií sa identifikuje oblasť návrhov, z ktorých sa vytvoria štvorce a pomocou spojovacej vrstvy RoI (Region of Interest - oblasť záujmu) [21] sa pretvoria na pevnú veľkosť, aby ich bolo možné vložiť do úplne prepojenej vrstvy. Z vektora funkcií RoI sa používa vrstva softmax na predpovedanie triedy navrhovanej oblasti a tiež hodnoty posunu pre ohraničujúci rámček. Dôvodom, prečo je „Fast R-CNN“ rýchlejší ako R-CNN je ten, že sa nemusia zakaždým privádzať návrhy oblastí do CNN. Namiesto toho sa konvolučná operácia vykoná pre každý obrázok iba raz a vygeneruje sa z nej mapa funkcií [22].



Obr. 1.8: Architektúra Fast R-CNN [22].

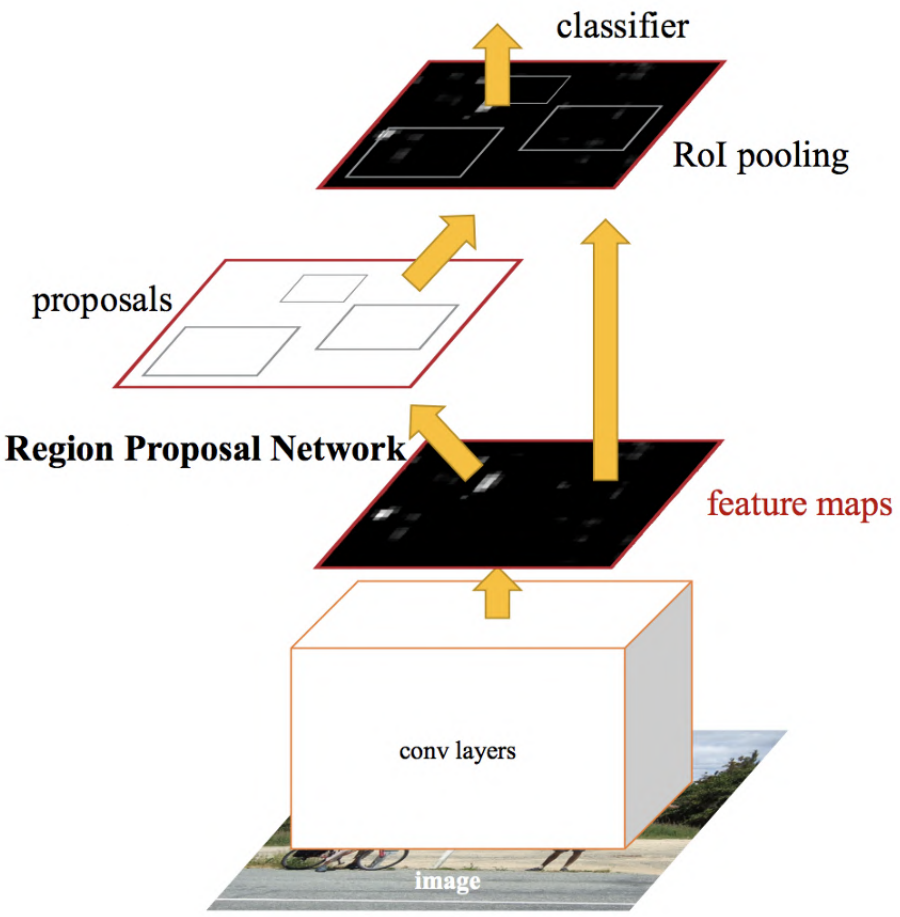
1.2.4 Faster R-CNN

Oba vyššie uvedené algoritmy (R-CNN (1.2.1) a Fast R-CNN (1.2.3)) používajú na vyhľadanie navrhovaných oblastí selektívne vyhľadávanie. Selektívne vyhľadávanie je pomalý a časovo náročný proces ovplyvňujúci výkon CNN. Preto Shaoqing Ren a jeho kolegovia prišli s algoritmom detekcie objektov Faster R-CNN (Rýchlejší R-CNN), ktorý eliminuje algoritmus selektívneho vyhľadávania a umožňuje sieti učiť sa návrhy regiónov. Podobne ako v prípade Fast R-CNN, je obraz vstupom do konvolučnej siete, ktorá poskytuje mapu konvolučných funkcií. Namiesto použitia algoritmu selektívneho vyhľadávania na mape funkcií pre identifikáciu navrhovaných oblastí sa na ich predpoveď používa samostatná sieť. Predpokladané návrhy oblastí sa potom pretvárajú pomocou združovacej vrstvy RoI, ktorá sa potom použije na klasifikáciu obrazu v navrhovanej oblasti a predpovedanie hodnôt posunu pre ohraničujúce rámčeky [23].



Obr. 1.9: Porovnanie testových rýchlostí nástrojov na rozpoznávanie objektov v obraze [14].

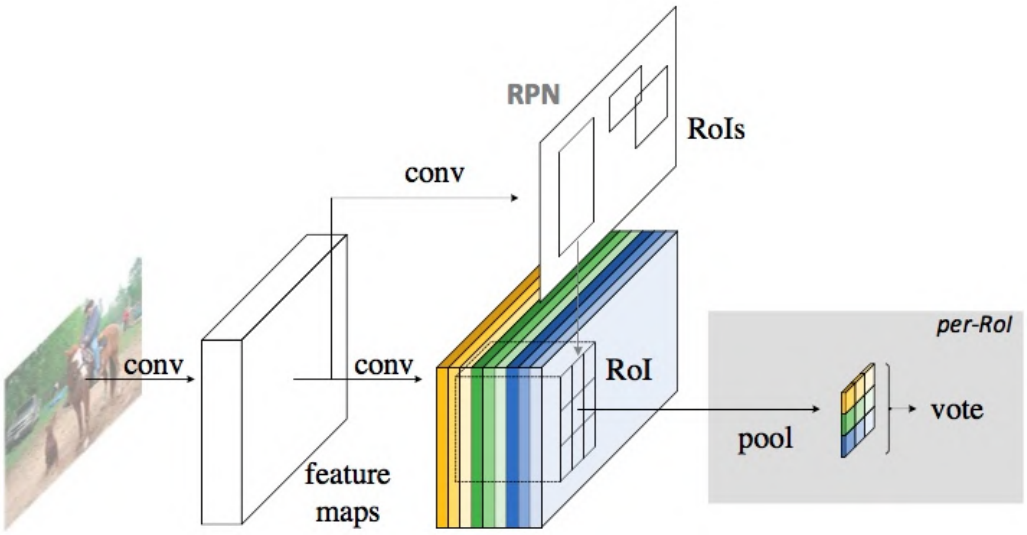
Na vyššie uvedenom obrázku 1.9 je znázornené porovnanie rýchlostí rozpoznávania objektov doposiaľ uvedených algoritmov. Faster R-CNN je už dostatočne rýchly na použitie v reálnom čase.



Obr. 1.10: Architektúra Faster R-CNN [23].

1.2.5 Region-based Fully Convolutional Network (R-FCN)

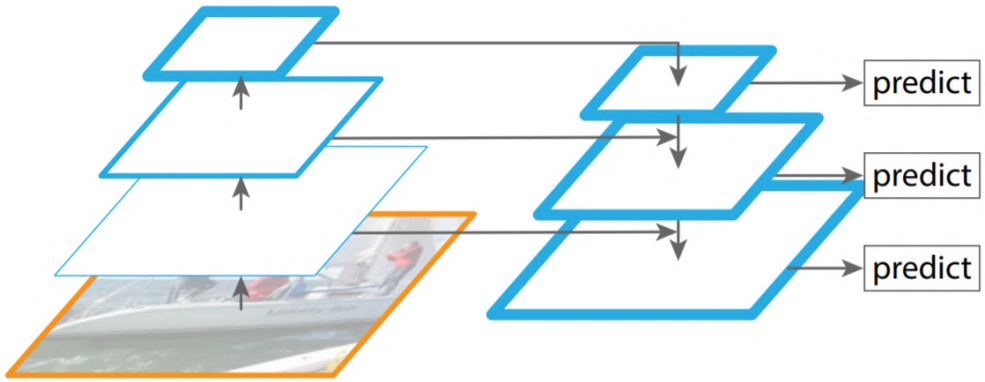
Regionálne založená plne konvolučná sieť R-FCN je detektor pre presnejšiu a efektívnejšiu detekciu objektov. Na rozdiel od predchádzajúcich detektorov oblastí ako Fast-RCNN (1.2.3) alebo Faster R-CNN (1.2.4), ktoré prácne používajú podsieť pre jednotlivé oblasti, je tento detektor úplne konvolučný a takmer všetky výpočty sú zdieľané na celom obrázku [24].



Obr. 1.11: Architektúra R-FN [24].

1.2.6 Feature Pyramid Networks (FPN)

Funkcia pyramídových sietí (FPN) používa metódu znázorňovania pyramíd, aj keď sa jej ostatné algoritmy vyhli, pretože je výpočtovo a pamäťovo náročná. Táto metóda ale využíva inherentné viacúrovňové pyramídové hierarchie hĺbkových konvolučných sietí na stavbu funkčných pyramíd s minimálnou náročnosťou. Pre server je vyvinutá architektúra s bočnými spojeniami pre budovanie vysoko-levelových funkčných máp vo všetkých mierkach. Táto architektúra ukazuje výrazné zlepšenie v extrakcií všeobecných funkcií v niekoľkých aplikáciách. Pri použití FPN v základnom systéme Faster R-CNN vykazuje táto metóda veľmi dobré výsledky [25].

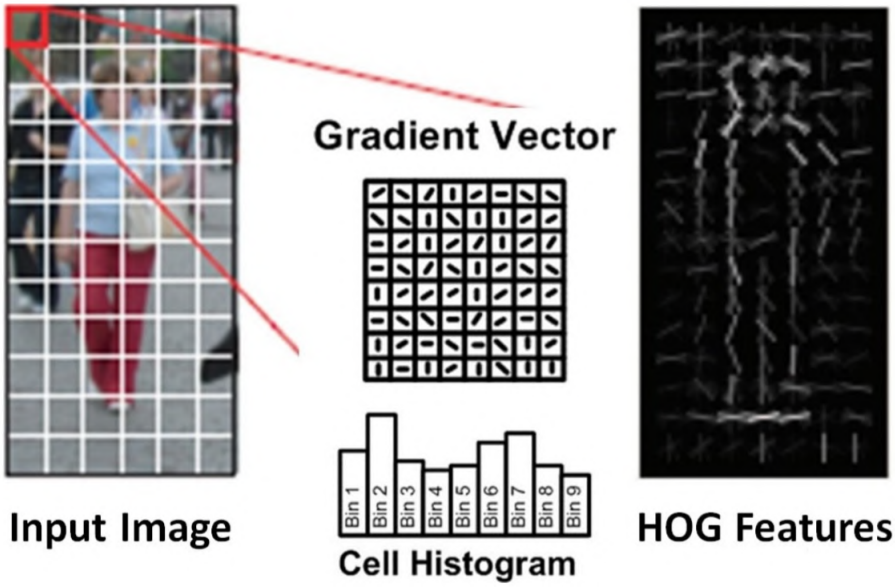


Obr. 1.12: Funkcia pyramídových sietí [25].

1.2.7 Histogram of Oriented Gradients (HOG)

Histogram of Oriented gradients (histogram orientovaných gradientov) je jedna z prvých metód rozpoznávania objektov v obraze, a nefunguje na princípe CNN. Táto metóda sa tiež pomenúva ako deskriptor črtov. Technika deskriptorov HOG počíta výskyty gradientovej orientácie v lokalizovaných častiach obrázka, ako je napríklad detekčné okno alebo určitá oblasť záujmu. Implementácia algoritmu deskriptoru HOG funguje nasledovne [26]:

1. Obrázok sa rozdelí do malých spojitých regiónov nazývaných bunky (cells) a pre jednotlivé pixely bunky sa vypočíta histogram gradientových smerov alebo okrajových orientácií.
2. Uskutoční sa diskretizácia každej bunky do uhlových zložiek (angular bins) v závislosti na orientácií gradientov.
3. Každý pixel bunky prispieva váženým gradientom do vzťahujúcej sa uhlovej zložky.
4. Skupiny susedných buniek sa považujú za priestorové oblasti nazývané bloky. Zoskupenie buniek do bloku je základom pre normalizáciu histogramov.
5. Normalizovaná skupina histogramov predstavuje blokový histogram. Sada týchto blokových histogramov predstavuje deskriptor.



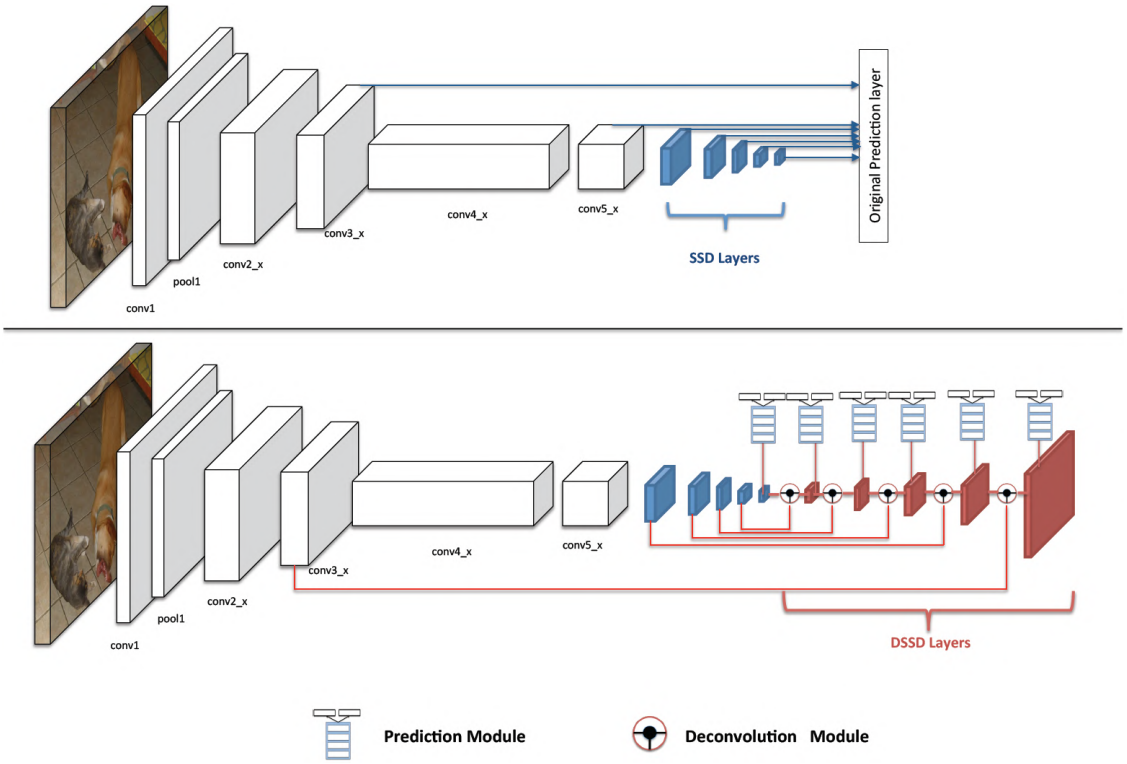
Obr. 1.13: Znázornenie extrakcie histogramovo orientovaných gradientov [27]

1.2.8 Single Shot Detector (SSD)

Single Shot Detector (detektor jedného záberu) je metóda na detekciu objektov v obrazoch pomocou jednej hĺbkovej neurónovej siete (single deep neural network). Prístup SSD diskretizuje výstupný priestor hraničných boxov (bounding boxes) do sady predvolených polí v rôznych pomeroch a mierkach podľa umiestnenia mapy funkcií (feature map). V čase predikcie neurónová sieť generuje skóre pre prítomnosť každej kategórie objektu v každom predvolenom políčku (default box) a prispôsobuje políčko, aby lepšie zodpovedalo tvaru objektu. Neurónová sieť navyše kombinuje predpovede z viacerých máp funkcií s rôznymi rozlíšeniami, aby prirodzene zvládla objekty rôznych veľkostí. SSD je jednoduchá v porovnaní s metódami, ktoré vyžadujú návrhy objektov, pretože úplne vylučuje generovanie návrhov a ich ďalšie fázy vzorkovania pixelov a spája všetky výpočty do jednej siete. Vďaka tomu sa SSD ľahko trénuje a integruje sa priamo do systémov, ktoré vyžadujú detekčný komponent [28]. Architektúra SSD je zobrazená na obrázku 1.14.

1.2.9 Deconvolutional Single Shot Detector (DSSD)

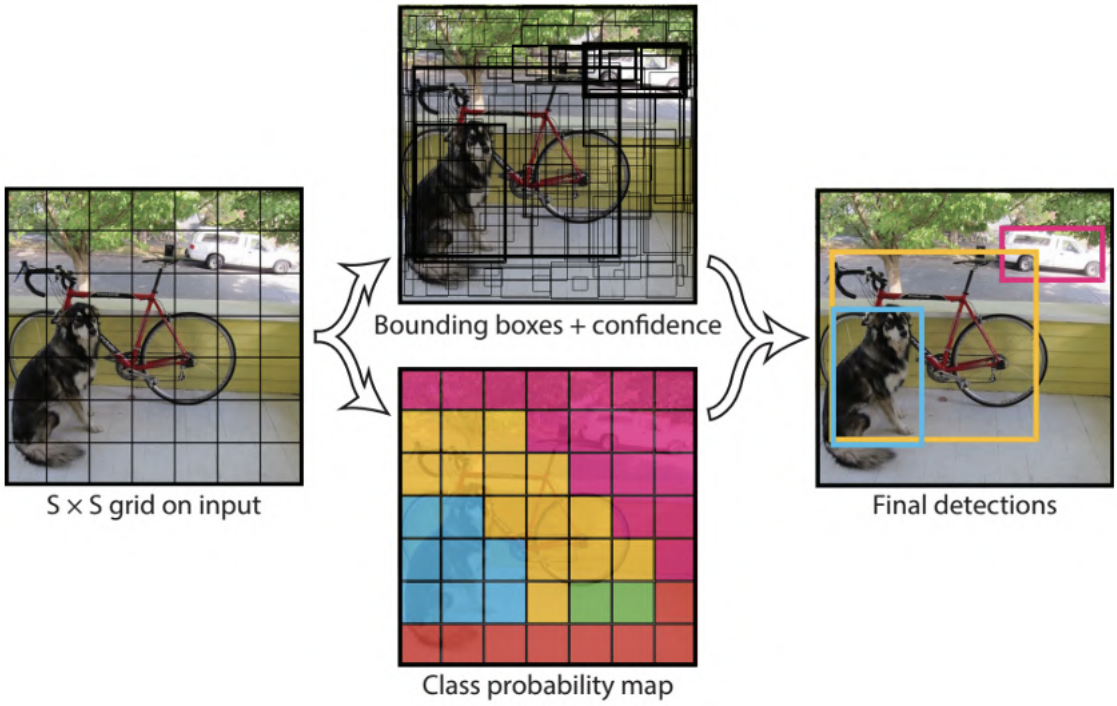
DSSD je vylepšená verzia SSD o dekonvolučné vrstvy, ktoré predstavujú ďalší rozsiahly kontext v rozpoznávaní objektov a vylepšujú presnosť, najmä pre malé objekty. Opatrným pridávaním ďalších fáz predtrénovaných transformácií, konkrétne modulu pre spätné väzby v dekonvolúcii a nového výstupného modulu, umožňuje tento nový prístup a vytvára potenciálnu cestu pre ďalší výskum rozpoznávania objektov [29].



Obr. 1.14: Architektúra SSD metódy (hore) a jej vylepšenie o dekonvolučné vrstvy DSSD (dole) [29].

1.2.10 YOLO (You Only Look Once)

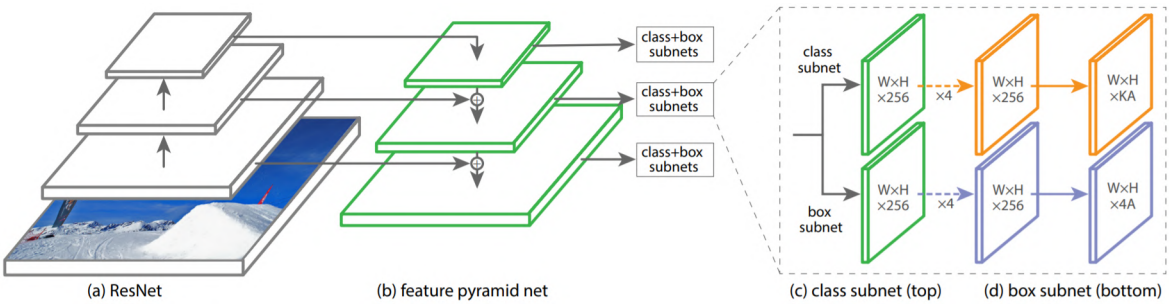
YOLO alebo You Only Look Once (pozrieš sa iba raz) je algoritmus podobný metóde SSD. V systéme YOLO sa používa iba jedna konvolučná sieť (single convolutional network), ktorá predpovedá ohraničujúce rámčeky a kategorické pravdepodobnosti pre tieto rámčeky. Funguje to tak, že YOLO si vezme vstupný obrázok a rozdelí ho do mriežky o veľkosti SxS. V rámci každej z mriežok vytvorí m ohraničujúcich rámčekov. Pre každý rámček sieť vygeneruje hodnoty pravdepodobnosti určitej kategórie a hodnoty posunu pre ohraničujúci rámček. Vyberú sa ohraničovacie rámčeky s pravdepodobnosťou vyššou ako je zadaná prahová hodnota pravdepodobnosti a použijú sa na lokalizáciu objektov v rámci obrázka. YOLO je rádovo rýchlejší algoritmus v porovnaní s inými a dokáže spracovať približne 45 snímok za sekundu vo verzii YOLOv3. Existuje aj verzia YOLOv3.tiny, ktorá je menej presná, ale má rýchlosť približne 155 snímok za sekundu [31]. Oficiálna stránka na stiahnutie je [32], ale nie je veľmi aktuálna, preto dnes už existuje aj verzia YOLOv4, ktorá je dostupná na [33], kde sú aktualizované aj staršie verzie. Nedávno bola zverejnená aj verzia YOLOv5 [34].



Obr. 1.15: Algoritmus YOLO [31].

1.2.11 RetinaNet

RetinaNet bola predstavená s cieľom vylepšiť nedostatky detektorov s jedným záberom, ako sú SSD a YOLO, a zároveň sa zaoberala triedami náročných pozadí a popredí obrázka. Táto metóda je navrhnutá tak, aby vyhovela Focal Loss (strata zaostrenia) metóde, ktorá sa snaží vyriešiť problém s nevyváženosťou tried obrázka a to tak, že priradí viac váhy (weights) ťažko alebo nesprávne klasifikovaným príkladom (objekty v pozadí s veľkým šumom). Viac informácií je na [35].

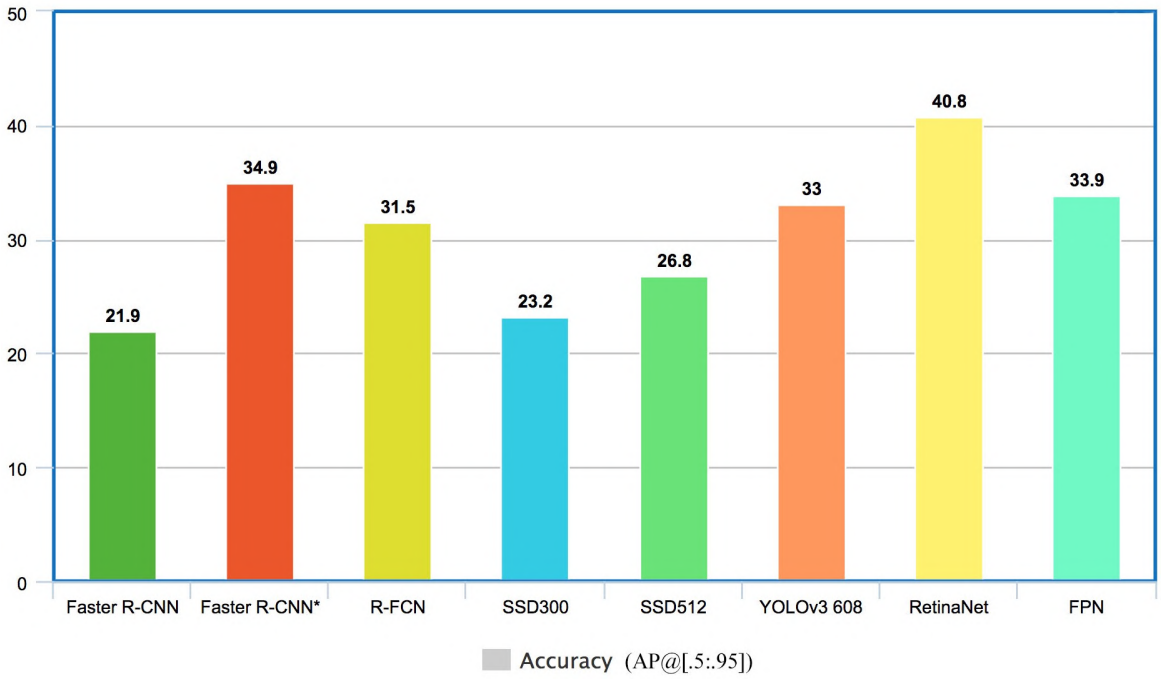


Obr. 1.16: Architektúra siete RetinaNet [35].

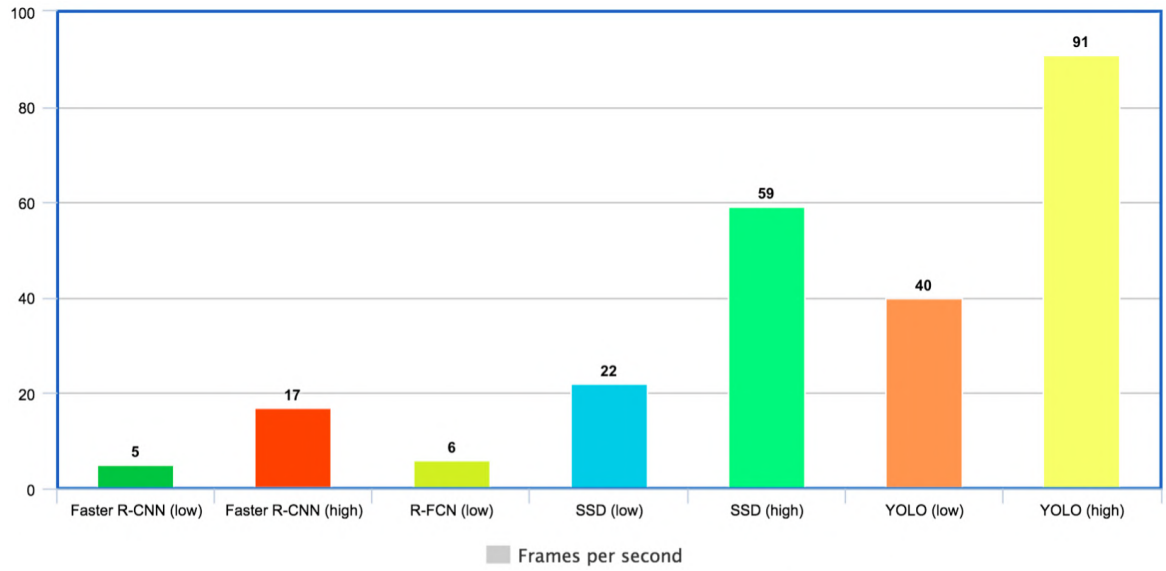
Klasifikačná podsieť predpovedá pravdepodobnosť prítomnosti objektu v konkrétnom umiestnení. Podsieť je akousi menšou verziou plne konvolučných sietí (FCN) pripojených ku každej úrovni pyramídových sietí (FPN). Mapa vstupných prvkov je prevzatá z danej úrovne pyramídy a štyroch 3x3 konvolučných vrstiev, nasleduje aktivácia ReLU (Rectified linear activation function – usmernená lineárna aktivačná funkcia) a ďalšia 3x3 konvolučná vrstva. Spolu s podsieťou klasifikácie je pripojená podsieť regresie poľa, aby sa anuloval posun od každého rámčeka k blízkemu hlavnému objektu. Umiestnenie negatívov alebo objektov na pozadí sa klasifikuje ako vektor obsahujúci iba nuly, zatiaľ čo pozitíva alebo popredie sa klasifikujú pomocou jedného ”horúceho” vektora. Za predpokladu, že predikcia je vektorom všetkých núl, ale cieľom bol vektor s jedným ”horúcim” faktorom (inými slovami falošne negatívny), sa strata zaostrenia vyhodnotí ako veľká hodnota pre túto oblasť.

1.2.12 Porovnanie nástrojov na rozpoznávanie objektov v obraze

Nie je rozumné porovnávať výsledky z rôznych článkov, pretože tieto experimenty sa uskutočňujú v rôznych prostrediach, ktoré nie sú určené na porovnanie. Napriek tomu sa ich Jonathan Hui vo svojom článku [36] pokúsil vykresliť spoločne, pre aspoň približné porovnanie.



Obr. 1.17: Porovnanie presností jednotlivých modelov [36].



Obr. 1.18: Porovnanie rýchlostí jednotlivých modelov [36].

1.3 Nástroje na rozpoznávanie hovorenej reči

Rozpoznávanie reči je interdisciplinárne odvetvie počítačovej vedy a výpočtovej lingvistiky, ktoré vyvíja metodiky a technológie, ktoré umožňujú rozpoznávanie a preklad hovoreného jazyka do textu pomocou počítačov. Je tiež známy ako automatické rozpoznávanie reči (ASR – Automatic Speech Recognition), počítačové rozpoznávanie reči alebo reč na text (STT – Speech To Text). Dnes už máme vytvorených mnoho knižníc a rôzne rozhrania na programovanie aplikácií (API – Application Programming Interface). V tejto časti sú popísané nástroje, ktoré sa môžu využiť pri programovaní funkcií na rozpoznávanie reči v jazyku python. Každý z nich je možné využiť pomocou knižnice `SpeechRecognition` 2.1.7 [37].

1.3.1 Google Speech-to-Text

Google Speech-to-Text je jeden z najpopulárnejších a najlepších API využívajúci pokročilé algoritmy neurónovej siete hĺbkového učenia spoločnosti Google. Podporuje rozpoznávanie hlasu pre viac ako 125 jazykov. Jedným z dôvodov, prečo je taký presný, je ten, že má možnosť výberu medzi jednotlivými modelmi strojového učenia (machine learning models) v závislosti od toho, na akú aplikáciu sa používa. Umožňuje rozpoznávanie reči v reálnom čase, kde sa spracováva audio z mikrofónu alebo vopred zaznamenaného zvukového súboru. Rozpoznávanie reči sa dá prispôsobiť tak, aby sa prepisovali výrazy a vzácne slová špecifické pre doménu. Ponúka aj rozšírené nastavenia interpunkcie, za účelom užitočnejšieho prepisovania viet a zníženia interpunkčných problémov. V jazyku Python sa použije nastavením `speech_recognition.Recognizer()` na `recognize_google()` alebo `recognize_google_cloud()`(vyžaduje inštaláciu google-cloud-speech package). Viac informácií je dostupných na [38].

1.3.2 Microsoft Azure

Microsoft Azure ponúka viaceré kognitívne služby a jednou z nich je aj STT. Tento API funguje pre viac ako 85 jazykov a taktiež ponúka možnosť zmeny modelov, vytvorených pomocou neurónových sietí, na zlepšenie presnosti pre špecifickú doménu. Aplikácie používajú systém s názvom System.Speech.Recognition na prístup a rozširovanie tejto základnej technológie rozpoznávania reči definovaním algoritmov na identifikáciu konkrétnych fráz alebo slovných výrazov a riadením správania tejto rečovej infraštruktúry za behu. Tento `Recognizer()` sa nastaví pomocou `recognize_bing()`. Viac informácií je dostupných na [39].

1.3.3 Houndify

Technológia Houndify poskytuje vyššiu presnosť rozpoznávania prostredníctvom integrácie komponentov pochopenia prirodzeného jazyka (NLU – Natural Language Understanding) a umožňuje ASR systému, založenému na neurálnych sieťach, prepisovať zložitú reč s väčšou presnosťou a taktiež pomocou bohatšieho kontextu pre rozpoznávanie slov. Houndify má technológiu Speech-to-Meaning, čo v preklade znamená reč do významu, ktorá sleduje reč v reálnom čase, pričom rozumie významu, a to ešte predtým, ako používateľ dohovorí. Namiesto typického dvojkrokového procesu prepisu reči do textu (ASR) a následného prenosu tohto textu do modelu NLU, dokončí Houndify

obe tieto úlohy v jednom kroku a prinesie rýchlejšie a presnejšie výsledky. Podporuje viacero programovacích jazykov. Použitie v jazyku Python je možné pomocou `recognize_houndify()`. Viac informácií je na [40].

1.3.4 CMUSphinx

Nástroj CMUSphinx je navrhnutý špeciálne pre nízkorozpočtové platformy. Zameriava sa na vývoj praktických aplikácií, nie na výskum. Podporuje niekoľko jazykov, ako je americká angličtina, angličtina vo Veľkej Británii, francúzština, mandarínčina, nemčina, holandčina, ruština, a schopnosť vytvárať modely pre ostatné jazyky. Má širokú škálu nástrojov na mnohé účely súvisiace s rozpoznávaním reči (hľadanie kľúčových slov, zarovnanie, hodnotenie výslovnosti). CMUSphinx poskytuje Vosk toolkit, čo je voľne dostupný nástroj na rozpoznávanie reči bez internetového pripojenia. Modely Vosk sú malé (približne 50 Mb), ale poskytujú nepretržitý prepis veľkej slovnej zásoby, odozvu s nulovou latenciou pomocou streamovacieho API, rekonfigurovateľnú slovnú zásobu a identifikáciu reproduktora. Väzby na rozpoznávanie reči sú implementované pre rôzne programovacie jazyky ako Python, Java, Node.JS, C ++ a ďalšie. V pythone sa nastaví pomocou `recognize_sphinx()`. Vosk je škálovateľný od malých zariadení, akými sú Raspberry Pi alebo smartphone so systémom Android, až po veľké zostavy. Ďalšie informácie sú dohľadateľné na [41].

1.3.5 Wit

Wit kombinuje rôzne najmodernejšie techniky spracovania prirodzeného jazyka a niekoľko nástrojov na rozpoznávanie reči, aby sa dosiahla nízka latencia a vysoká odolnosť voči okolitému hluku a parafrázovým variáciám (existujú milióny spôsobov, ako povedať to isté). Snahou Wit je vytvorenie čo najjednoduchšieho prostredia pre vývojárov. Podporuje viac než 132 jazykov na rozpoznávanie reči. V Pythone sa použije pomocou `recognize_wit()`. Ďalšie informácie sú dostupné na [42].

1.3.6 IBM Watson

IBM Watson™ Speech to Text využíva strojové učenie na kombináciu znalostí gramatiky, jazykovej štruktúry a zloženia zvukových a hlasových signálov na presnú transkripciu ľudského hlasu. Prijímaním reči sa systém neustále aktualizuje a zdokonaľuje prepis reči. IBM Watson poskytuje API, vďaka ktorému je vhodný pre každú aplikáciu, kde je vstupom reč a výstupom textový prepis. Môže byť použitý pre aplikácie ako sú hlasovo automatizované chatboty, analytické nástroje pre call centrá zákazníckych služieb a prepis multimédií. Možnými aplikáciami sú okrem iných aj hlasové ovládanie zabudovaných zariadení, prepisovanie stretnutí a konferenčných hovorov a diktovanie správ a poznámok. Umožňuje rýchly a presný prepis reči vo viac než 11 jazykoch. V Pythone sa použije pomocou `recognize_ibm()`. Viac informácií je na [43].

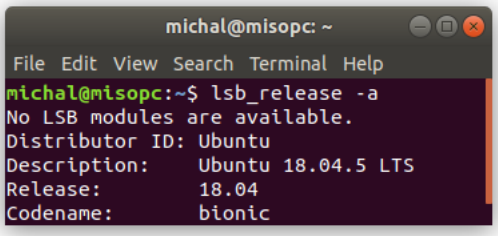
2 Praktická časť

Táto časť sa zaoberá konkrétnym vytvorením modelu robotickej hlavy a jej naprogramovaním. Ako prvé tu sú spomenuté všetky použité nástroje a knižnice pre programovací jazyk Python. Ďalšia kapitola sa zaoberá použitou elektronikou a konštrukciou robotickej hlavy. Na záver sú popísané všetky vytvorené funkcie robotickej hlavy na interakciu s človekom a ich názorné ukážky.

2.1 Použité nástroje a ich inštalácie

2.1.1 Operačný systém

Ako operačný systém pre splnenie tejto bakalárskej práce bol zvolený Linux, presnejšie Ubuntu 18.04. Výhodou tohto operačného systému je, že nie je licencovaný a je voľne dostupný na [44]. S týmto systémom sa bolo potrebné zoznámiť, avšak celková práca a inštalácia knižníc bola jednoduchšia v porovnaní s inými operačnými systémami. V čase inštalácie bola dostupná aj nová verzia Ubuntu 20.04, no nepodporovala všetky nástroje, ktoré bolo potrebné použiť, preto bola použitá verzia 18.04 LTS Bionic Beaver (viď obr. 2.1).



Obr. 2.1: Verzia Ubuntu v termináli.

2.1.2 Kamera Intel RealSense D435

Pre tento projekt bola zvolená kamera Intel RealSense D435, ktorá zaznamenáva RGB obraz, ale aj hĺbkovú mapu, obsahujúcu informácie o vzdialenostiach objektov od kamery. Táto kamera má široké zorné pole a je vhodná na programovanie aplikácií pre roboty. Na použitie tejto kamery je potrebná inštalácia RealSense SDK 2.0, ktorá je popísaná na [45]. Využitie tejto kamery pomocou jazyka Python je umožnené inštaláciou knižnice `pyrealsense2`, uvedenou na [46].

Obr. 2.2: Intel RealSense D435.

2.1.3 CUDA a cuDNN

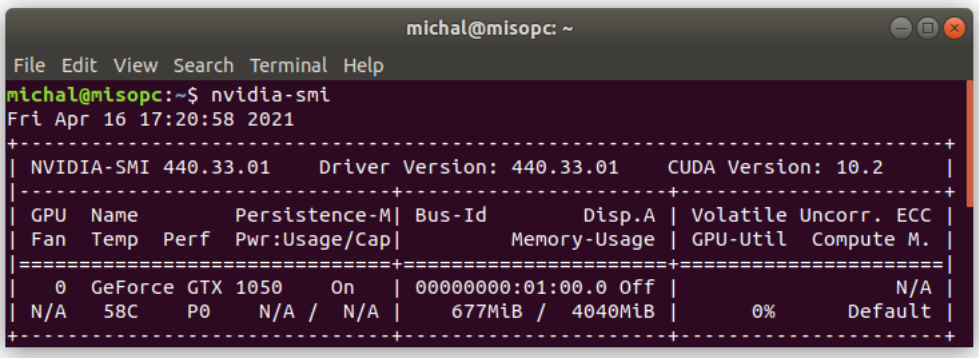
Tieto nástroje boli nainštalované pre rýchlejšiu prácu s neurónovými sieťami využitím grafickej procesorovej jednotky (GPU), pretože s procesorom základnej jednotky (CPU), by nebola možná detekcia objektov v reálnom čase.

CUDA

CUDA® je paralelná výpočtová platforma a programovací model vyvinutý spoločnosťou NVIDIA pre všeobecné výpočty na GPU. CUDA umožňuje dramaticky zrýchliť výpočtové aplikácie využitím sily GPU. Pre grafickú kartu GeForce GTX 1050 v použitom notebooku bola nainštalovaná verzia Cuda Toolkit 10.2. Podrobná inštalácia je uvedená na [47].

cuDNN

NVIDIA CUDA® Deep Neural Network (cuDNN) je GPU-akcelerovaná knižnica pre hĺbkové neurónové siete. Knižnica cuDNN poskytuje vysoko vyladené implementácie pre štandardné rutiny, napríklad pre doprednú a spätnú konvolúciu, združovanie (pooling), normalizáciu a aktivačné vrstvy. Na stiahnutie tejto knižnice je potrebná registrácia do NVIDIA Developer Programu. Je nutné zvoliť verziu cuDNN rovnakú ako verzia CUDA. Podrobná inštalácia je na [48].



Obr. 2.3: Verzia CUDA v termináli Ubuntu.

2.1.4 Tesseract OCR

Tesseract OCR engine je nástroj založený na neurónovej sieti, ktorý je zameraný na rozpoznávanie znakov v obraze. V tejto práci bol použitý na možnosť čítania textov pomocou kamery. Dokumentácia je uvedená na [49].

2.1.5 OpenCV

OpenCV (Open Source Computer Vision Library) je softvérová knižnica obsahujúca množstvo algoritmov na počítačové videnie a strojové učenie. OpenCV bol v tomto prípade použitý ako nástroj na komunikáciu s konvolučnou neurónovou sieťou a inými modelmi, nastavenie obrazu a vyznačenie oblastí záujmov. Na inštaláciu je potrebná tzv. build from source inštalácia s využitím programu cmake, do ktorého boli zadané požiadavky systému a nástroje ako CUDA (2.1.3) a tesseract (2.1.4), s ktorými chceme, aby OpenCV spolupracoval. Príkladná inštalácia je uvedená na [50]. Potrebné zložky na stiahnutie sú k dispozícii na [51] [52].

2.1.6 YOLO

Pre túto bakalársku prácu bol použitý nástroj YOLO, ktorý je pospísaný v kapitole 1.2.10, a to pre jeho rýchlosť rozpoznávania, ktorá umožnila detekciu objektov v reálnom čase. Výhodou bola aj jednoduchá implementácia pomocou jazyka Python. Bol využitý model yolov3 [33], ktorý dosahoval najlepšie výsledky v pomere rýchlosti a presnosti rozpoznávania.

2.1.7 SpeechRecognition

SpeechRecognition je knižnica na vykonávanie rozpoznávania reči s podporou niekoľkých nástrojov a rozhraní API, online aj offline. V tejto práci bola využitá s algoritmom rozpoznávania reči od google (1.3.1) na ovládanie a komunikáciu s robohlavou. Táto knižnica je popísaná na [37].

2.1.8 WolframAlpha

Wolfram Alpha je výpočtový vyhľadávací nástroj (niekedy označovaný ako „odpoveďový modul“). Rozhranie vyzerá podobne ako bežné vyhľadávacie nástroje, ale výrazy zadané do vyhľadávacieho poľa vedú skôr k odpovediam na konkrétne otázky ako na zoznamy webových stránok, ktoré môžu byť pre daný dopyt relevantné. Ponúka API, ktorý je možné využiť vo viacerých programovacích jazykoch, avšak funguje len v angličtine. Viac informácií je na [53].

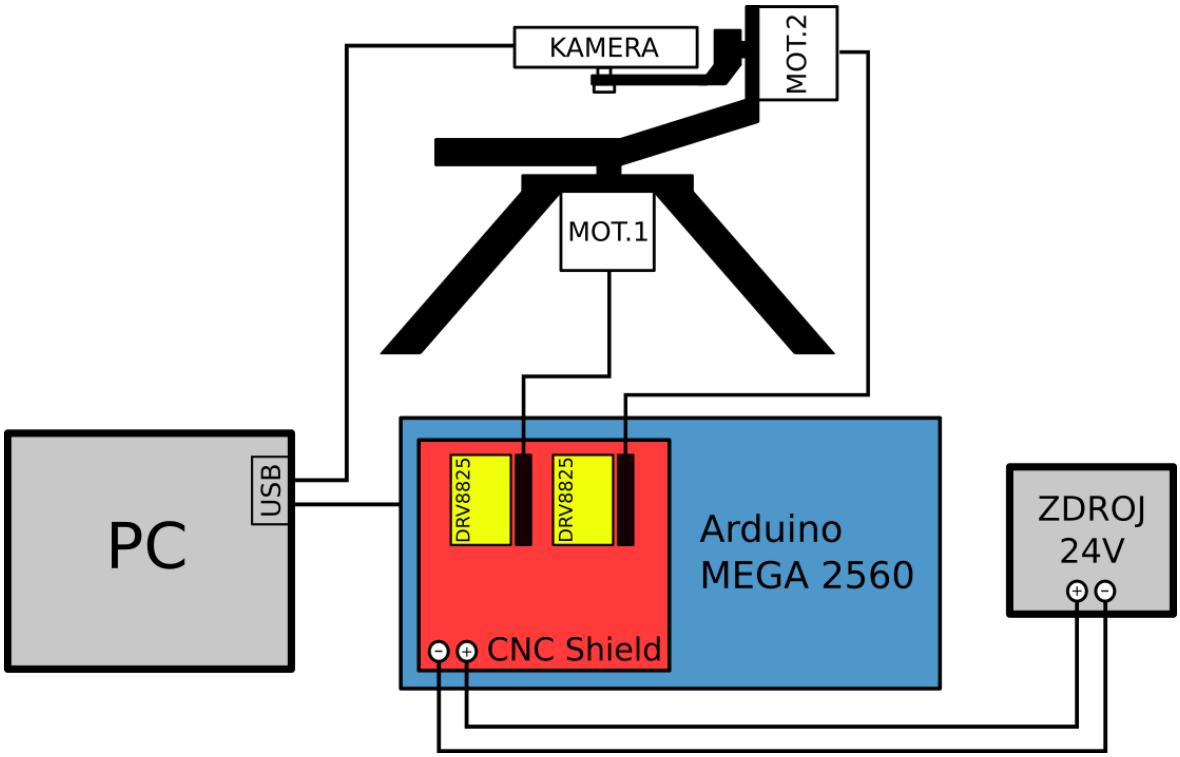
2.1.9 Text To Speech (STT)

Využité boli rôzne nástroje na preklad textu do reči. Asi najlepší z použitých nástrojov je pyttsx3 [55], no nepodporuje jazyky ako čeština alebo slovenčina. Preto boli použité nástroje ako espeak-ng [54] a gTTS (google Text-To-Speech) [56], ktoré možno použiť aj s inými jazykmi.

2.2 Zostavenie robotickej hlavy

2.2.1 Elektronika robotickej hlavy

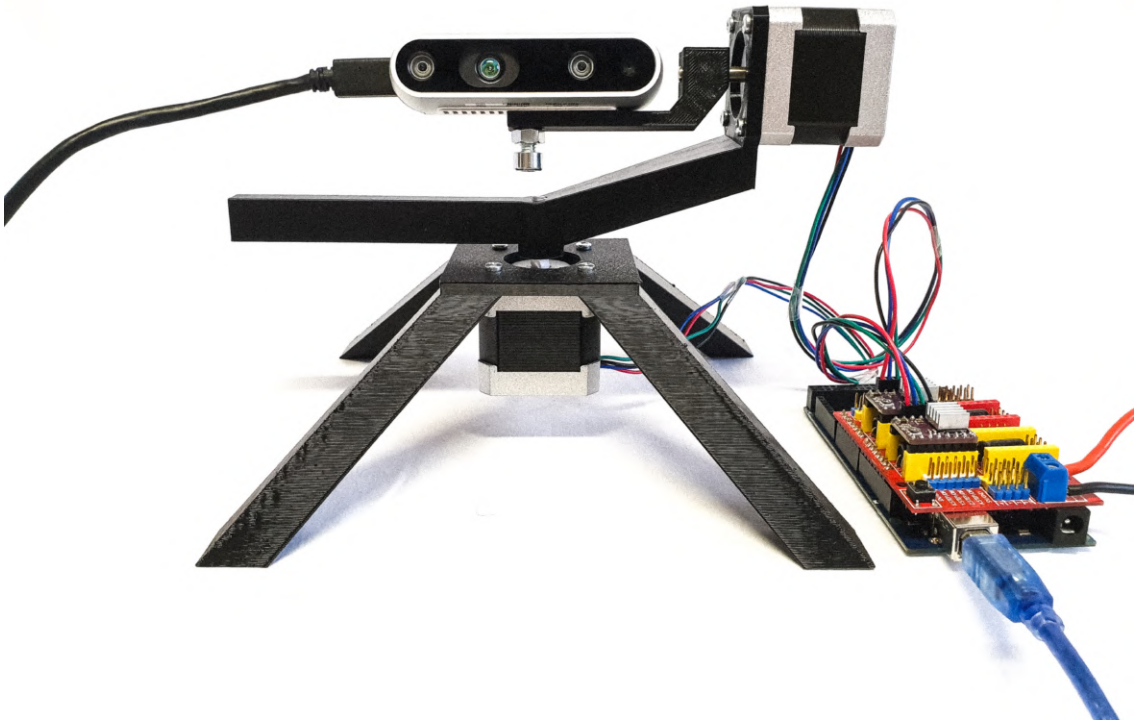
Na umožnenie pohybu robotickej hlavy boli použité dva krokové motory uvedené na stránke [57]. Jednotlivé motory boli spojazdnené pomocou driveru DRV8825 uvedeného na stránke [58], s ktorým sa nastavilo potrebné napájanie motora. Motory a drivery boli pripojené do CNC shieldu, ktorý je napájaný z 24-voltového zdroja. Tento shield bol pripevnený na Arduino MEGA 2560 [59]. Výsledné zapojenie je možné vidieť na obr. 2.4. Do Arduina bol nahratý kód umožňujúci posielanie správy z Pythonu, ktorá obsahuje informáciu na pohyb motora 1 a motora 2. Tento kód bol prevzatý z predošlej verzie robohlavy a následne modifikovaný pre jej nový dizajn.



Obr. 2.4: Blokové schéma.

2.2.2 Konštrukcia robotickej hlavy

Z dôvodu nepriaznivých podmienok prístupu do priestorov školy a zjednodušenia vypracovania tejto bakalárskej práce bola navrhnutá nová provizórna konštrukcia robohlavy. Táto konštrukcia bola namodelovaná v študentskej verzii programu inventor a následne vytlačená na 3D tlačiarni (viď obr. 2.5)



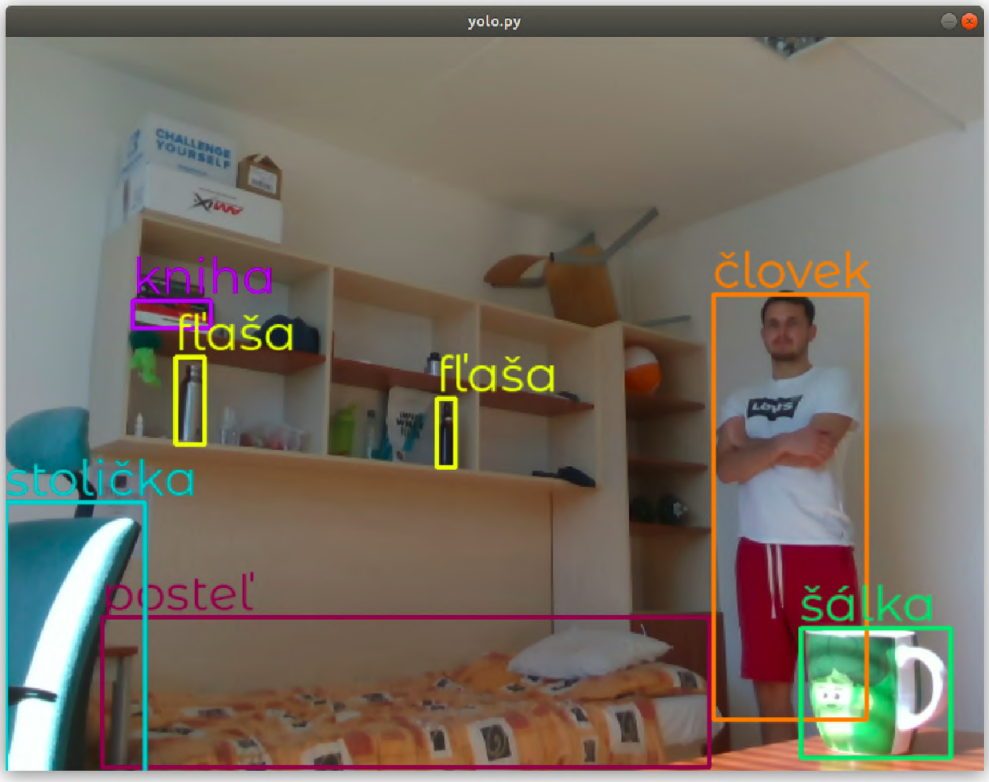
Obr. 2.5: Výsledný model robohlavy.

2.3 Interaktívne hry s užívateľom

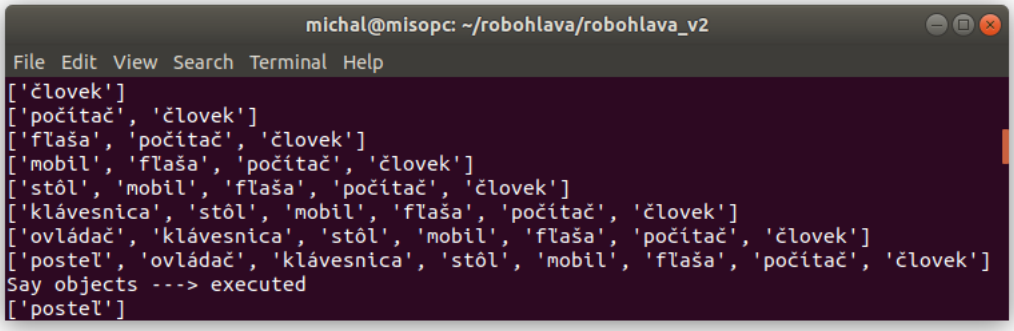
V tejto časti sú popísané jednotlivé funkcie robohlavy predstavujúce interaktívne hry s užívateľom. Všetky tieto funkcie sú napísané v jazyku Python verzie 3.6.9. Pri ich programovaní boli využité funkcie predošlej robohlavy, a to `camera.py`, ktorá umožňuje získanie obrazu z kamery RealSense (2.1.2), a funkcia `person_class.py`, ktorá obsahuje algoritmus na priradenie ID sledujúcemu človeku vo funkcií `track.py` (2.3.2).

2.3.1 yolo.py

Funkcia `yolo.py` využíva predtrénovanú sieť yolov3 (1.2.10) na rozpoznávanie objektov v obraze v reálnom čase. Dokáže rozlíšiť 80 rôznych predmetov, ktoré boli preložené do slovenčiny, a každému z nich bola priradená jedinečná farba rámčeka. Cieľom tejto hry je ukázať robohlave určitý počet rôznych predmetov, v tomto prípade 8, ktoré ona následne vysloví. Rozpoznané predmety sa ukladajú do listu, ktorý je vypísaný pomocou terminálu po rozpoznaní nového objektu.



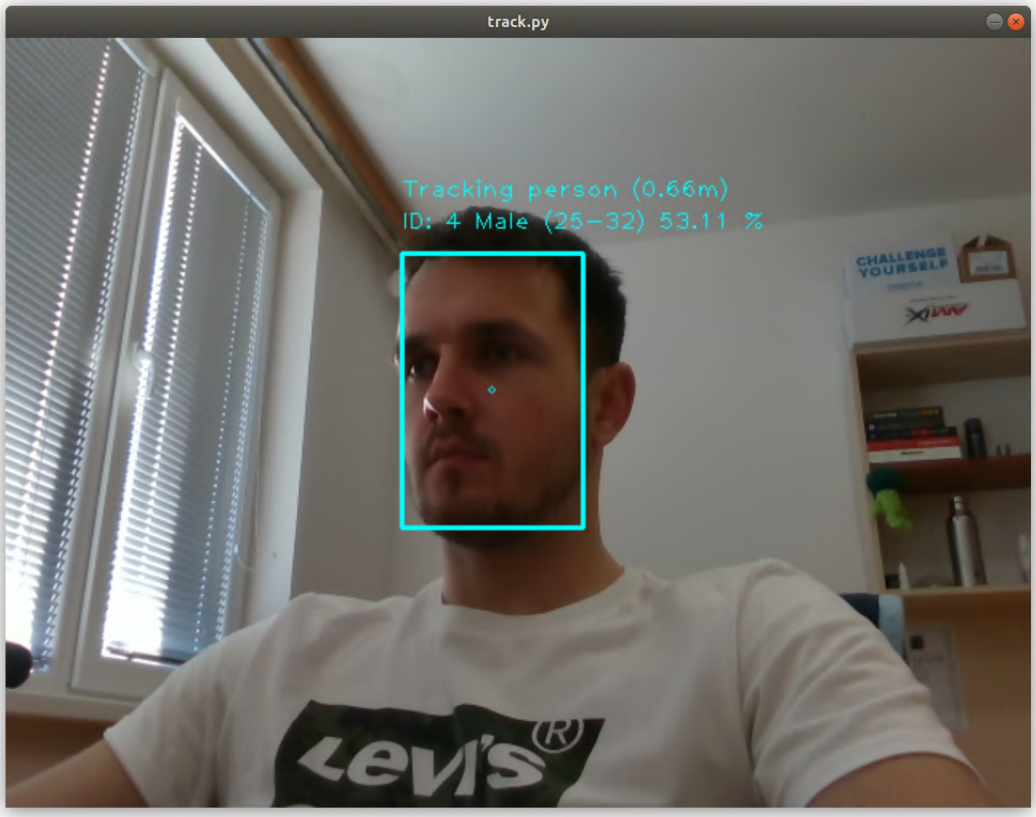
Obr. 2.6: Obraz z funkcie yolo.py.



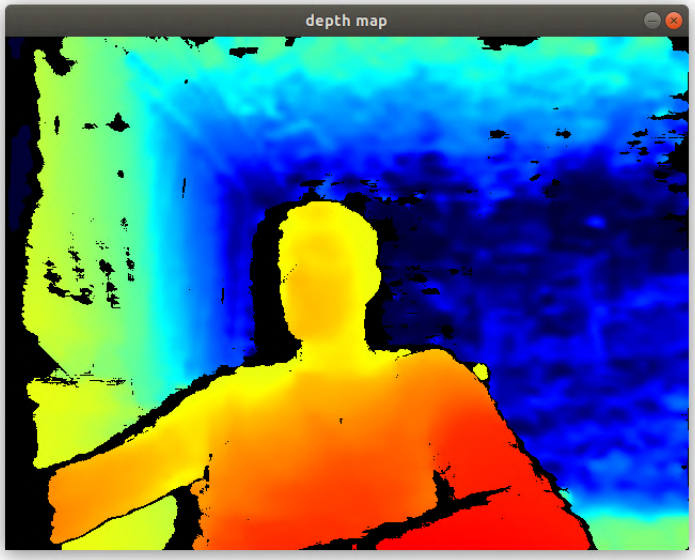
Obr. 2.7: Terminál z funkcie yolo.py.

2.3.2 track.py

Funkcia `track.py` umožňuje robohlave trackovanie tváre v obraze. Využíva viacero menších modelov na rozpoznávanie tváre, veku a pohlavia. V tomto prípade je použitá aj hĺbková mapa kamery RealSense, z ktorej získava informáciu o vzdialenosti od sledovanej osoby. V situácii, keď sa v obraze nachádza viacero osôb, sleduje robohlava osobu, ktorá bola v obraze skôr (má nižšiu ID z funkcie `person_class.py`). Ako to celé funguje, je možné vidieť na obr. 2.8 a hĺbkovú mapu na obr. 2.9. Pri tejto funkcii je využité Arduino, do ktorého je poslaná informácia pre pohyb motorov, ktoré otočia robohlavu tak, aby bol stred rámčeka (centroid) približne v strede obrazu.



Obr. 2.8: Obraz z funkcie track.py.

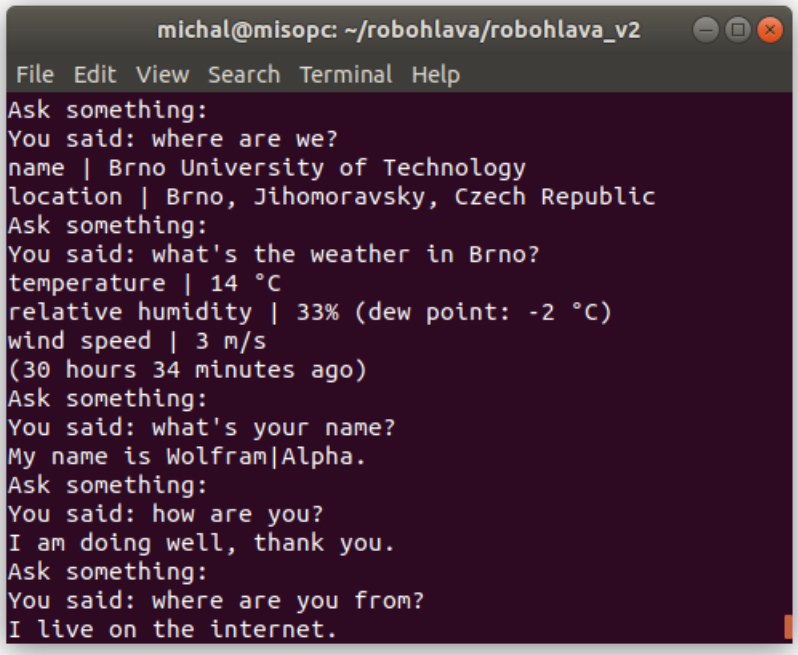


Obr. 2.9: Hĺbková mapa z funkcie track.py.

2.3.3 jarvis.py

Funkcia `jarvis.py` odpovedá na otázky užívateľa využitím WolframAlpha API 2.1.9. Na rozpoznanie hovorenej reči bola využitá knižnica `SpeechRecognition` a následne bol použitý rozpoznávač `recognize_google()`. Tento recognizer dokáže rozpoznať slo-

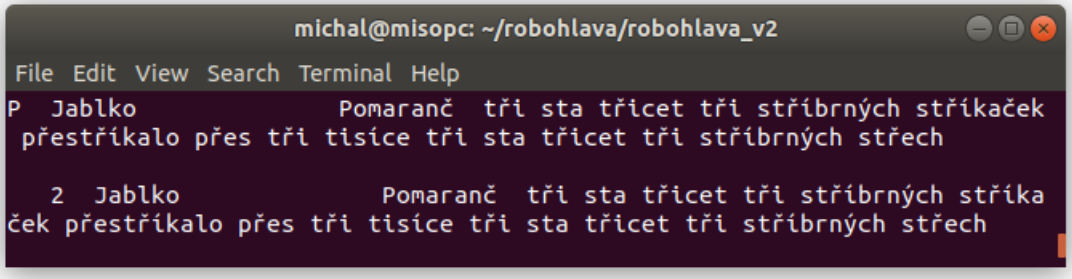
venský aj český jazyk, avšak WolframAlpha funguje len v angličtine. Po spustení funkcie robohlava čaká na otázku, ktorú následne rozpozná, pošle do WolframAlpha a odpovie, ak pozná odpoveď. Všetky kroky sú postupne zobrazované v termináli ako na obr. 2.10.



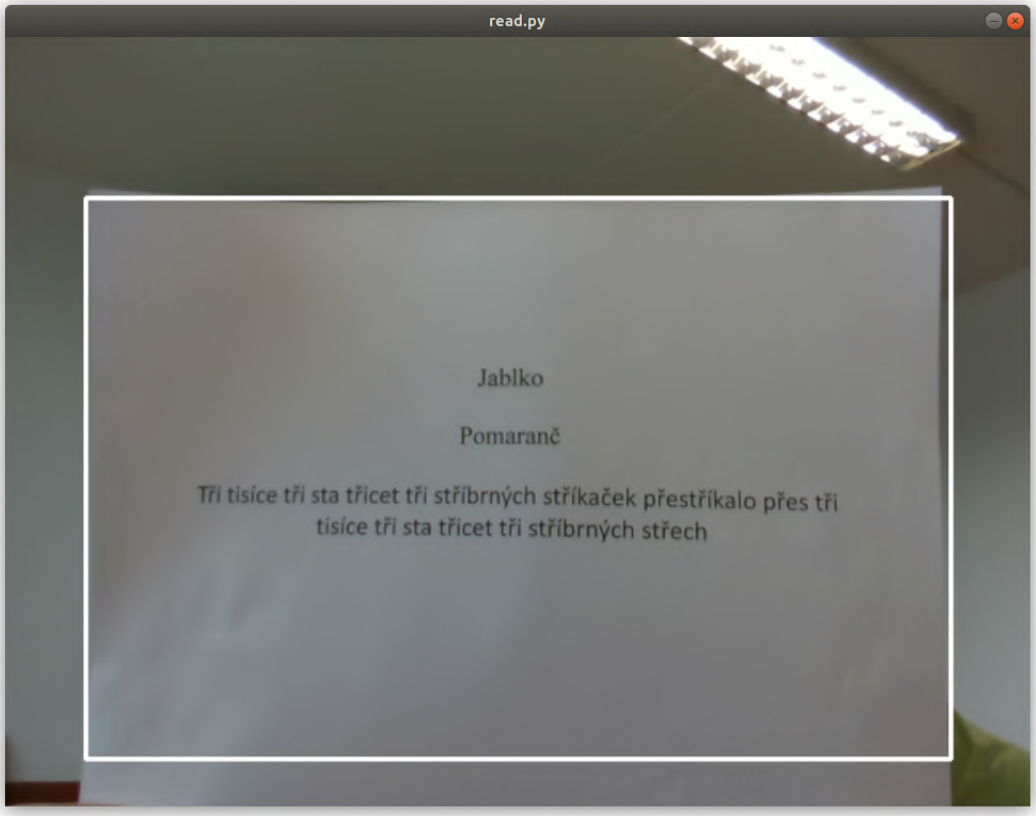
Obr. 2.10: Terminál z funkcie jarvis.py.

2.3.4 read.py

Funkcia `read.py` umožňuje robohlave prečítať text z papiera v reálnom čase. Rozpoznávanie textu z obrázka je umožnené nástrojom Tesseract-OCR (2.1.4) pre český jazyk. V tejto funkcii nie sú naprogramované žiadne algoritmy na rozdelenie textu do jednotlivých slov a následné vyslovenie prečítaného textu. Funguje len rozpoznanie textu z obrázka (obr. 2.12) a vypísanie znakov do terminálu, ako je to zobrazené na obr. 2.11.



Obr. 2.11: Terminál pri funkcii read.py.



Obr. 2.12: Obraz z kamery pri funkcii read.py.

2.3.5 repeat.py

Funkcia `repeat.py` je zjednodušená funkcia `jarvis.py`, ktorá namiesto hľadania odpovede na otázku rozpoznanú reč povedanú užívateľom rovno vysloví. Táto funkcia bola vytvorená na testovanie robohlavy zamerané na možnosti rozpoznávania rôznych jazykov a komplexnosť viet.

Záver

Cieľom tejto bakalárskej práce bolo vytvorenie modelu robotickej hlavy a naprogramovanie jednoduchých interaktívnych hier s užívateľom.

V prvej časti sme rozdelili robotické hlavy do troch základných skupín podľa ich vzhľadu a správania sa. Následne sme si vysvetlili problematiku návrhu ako hardwaru, tak aj softwaru robotických hláv. V ďalšej časti sme dohľadali dostupné nástroje na rozpoznávanie objektov v obraze. Tieto nástroje sme stručne popísali a ukázali, ako vyzerá ich architektúra. Napokon sme definovali dostupné nástroje na rozpoznávanie hovorenej reči, ktoré môžeme využiť pri programovaní v jazyku Python.

V praktickej časti sme si na úvod popísali všetky použité nástroje, ktoré sme využili pri programovaní modelu interaktívnej robotickej hlavy. Následne sme uviedli použitú elektroniku, na základe ktorej sme navrhli model robotickej hlavy. Na záver sme popísali naprogramované funkcie.

Výsledný model robohlavy dokáže pomocou obrazu získaného z kamery Intel RealSense D435 a pomocou modelu YOLO rozpoznávať objekty v reálnom čase s rýchlosťou 9 snímok za sekundu. Ďalej dokáže pomocou menšieho modelu na rozpoznávanie tváre a pomocou motorov poháňaných arduinom, trackovať tvár človeka s rýchlosťou približne 30 snímok za sekundu. Robohlava je schopná rozpoznať hovorenú reč v angličtine, ale aj v slovenčine pomocou nástroja od spoločnosti Google. Na anglické otázky ako „where are we?“ alebo „how much is 5 + 5?“ dokáže odpovedať pomocou API od WolframAlpha. Navyše dokáže pomocou nástroja Tesseract OCR rozpoznať text a následne ho vypísať v termináli Ubuntu.

V tejto bakalárskej práci boli splnené všetky ciele až na to, že sme neprevzali a nezostavili predom navrhnutý model robotickej hlavy, ale navrhli sme zjednodušený model, ktorý bol následne vytlačený pomocou 3D tlačiarne. V prípade ďalšieho vývoja interaktívnej robotickej hlavy, by bolo dobré navrhnúť konštrukciu so 6 stupňami voľnosti, aby bol umožnený náklon do strán. Ďalej by bolo vhodné implementovať do tohto návrhu pohyblivé oči, ktoré by výrazne zmenili vzhľad robohlavy, a taktiež jednoduché komponenty na mimiku tváre. Čo sa týka softwaru, bolo by určite vhodné spojiť všetky tieto naprogramované funkcie pomocou stavového stroja (state machine), s ktorým by sa umožnilo napríklad hlasové ovládanie alebo by sa vytvorilo grafické užívateľské rozhranie na výber funkcie. Tento stavový model sme sa pokúsili vytvoriť, no zatiaľ neboli dosiahnuté zodpovedajúce výsledky.

Literatúra

[1] MUTHUGALA, Mavj, M K B S MUNASINGHE, M H C LAKSHAN, L H H MADURANGI a A G B P JAYASEKARA. Design of an interactive robotic head with human-like movements. In: 2013 IEEE 8th International Conference on Industrial and Information Systems [online]. IEEE, 2013, 2013, s. 355-360 [cit. 2021-03-19]. ISBN 978-1-4799-0910-0. Dostupné z: https://www.researchgate.net/publication/271557048_Design_of_an_interactive_robotic_head_with_human-like_movements

[2] Sophia. *Hanson Robotics* [online]. [cit. 2021-03-20]. Dostupné z: https://www.hansonrobotics.com/sophia/

[3] Aplied Informatics Group. Flobi. *University of Belfield* [online]. [cit. 2021-03-20]. Dostupné z: https://aiweb.techfak.uni-bielefeld.de/flobi-head

[4] VAN BREEMEN, Albert, Xue YAN a Bernt MEERBEEK. ICat. In: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems - AAMAS '05 [online]. New York, New York, USA: ACM Press, 2005, 2005, s. 143- [cit. 2021-03-20]. ISBN 1595930930. Dostupné z: https://www.researchgate.net/publication/221454739_iCat_an_animated_user-interface_robot_with_personality

[5] Probo. *Vrije Universiteit Brussel* [online]. [cit. 2021-03-20]. Dostupné z: http://probo.vub.ac.be/

[6] PR2. *Willow Garage* [online]. [cit. 2021-03-20]. Dostupné z: http://www.willowgarage.com/pages/pr2/overview

[7] Cervical vertebrae. *Wikipedia* [online]. [cit. 2021-03-20]. Dostupné z: https://en.wikipedia.org/wiki/Cervical_vertebrae

[8] Head-neck-movement. *HEAVYWEIGHTBJJ.COM* [online]. [cit. 2021-03-20]. Dostupné z: https://heavyweightbjj.com/2016/03/13/protect-yo-neck/head-neck-movement/

[9] Cervical Spine Functional Anatomy and the Biomechanics of Injury Due to Compressive Loading. *PMC, US National Library of Medicine, National Institutes of Health* [online]. [cit. 2021-03-20]. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1250253/

[10] Eye movement. *Wikipedia* [online]. [cit. 2021-03-20]. Dostupné z: https://en.wikipedia.org/wiki/Eye_movement

[11] Normal Range of Eye Movement and Its Relationship to Age. *Investigative Ophthalmology & Visual Science* [online]. [cit. 2021-03-20]. Dostupné z: https://iovs.arvojournals.org/article.aspx?articleid=2638575

[12] KISHI, Tatsuhiro, Kenji HASHIMOTO a Atsuo TAKANISHI. Human Like Face and Head Mechanism. GOSWAMI, Ambarish a Prahlad VADAKKEPAT, ed. Humanoid Robotics: A Reference [online]. Dordrecht: Springer Netherlands, 2016, 2017-09-18, s. 1-26 [cit. 2021-04-10]. ISBN 978-94-007-7194-9. Dostupné z: https://doi.org/10.1007/978-94-007-7194-9_89-1

[13] BALDWIN, Elizabeth A., Jinhe BAI, Anne PLOTTO a Sharon DEA. Electronic Noses and Tongues: Applications for the Food and Pharmaceutical Industries. Sensors [online]. 2011, 11(5), 4744-4766 [cit. 2021-03-30]. ISSN 1424-8220. Dostupné z: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3231405/

[14] R-CNN, Fast R-CNN, Faster R-CNN, YOLO — Object Detection Algorithms. *towards data science* [online]. [cit. 2021-04-01]. Dostupné z: https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e

[15] A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way. *towards data science* [online]. [cit. 2021-04-01]. Dostupné z: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[16] CAMACHO, Cezanne. Convolutional Neural Networks. *Github* [online] [cit. 2021-04-11]. Dostupné z: https://cezannec.github.io/Convolutional_Neural_Networks/

[17] GIRSCHICK, Ross, Jeff DONAHUE, Trevor DARRELL, Jitendra MALIK. Rich feature hierarchies for accurate object detection and semantic segmentation, Tech report (v5). *UC Berkeley*[online]. [cit. 2021-03-31]. Dostupné z: https://arxiv.org/pdf/1311.2524.pdf

[18] UIJLINGS, Jasper R.R., K.E.A. VAN DE SANDE, T. GEVERS, and A.W.M. SMEULDERS. Selective Search for Object Recognition. *University of Trento (Italy), University of Amsterdam (the Netherlands)* [online]. [cit. 2021-03-31]. Dostupné z: https://ivi.fnwi.uva.nl/isis/publications/2013/UijlingsIJCV2013/UijlingsIJCV2013.pdf

[19] Support Vector Machine — Introduction to Machine Learning Algorithms. *towards data science* [online]. [cit. 2021-04-01]. Dostupné z: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[20] HE, Kaiming, Xiangyu ZHANG, Shaoqing REN, and Jian SUN. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. [online]. [cit. 2021-03-31]. Dostupné z: https://arxiv.org/pdf/1406.4729.pdf

[21] LIN, Huibao, Jennie SI a Glen P. ABOUSLEMAN. Region-of-Interest Detection and its Application to Image Segmentation and Compression. In: 2007 International Conference on Integration of Knowledge Intensive Multi-Agent Systems

[online]. IEEE, 2007, 2007, s. 306-311 [cit. 2021-04-10]. ISBN 1-4244-0944-6. Dostupné z: https://ieeexplore.ieee.org/document/4227566

[22] GIRSHICK, Ross. Fast R-CNN. *Microsoft Research* [online]. [cit. 2021-04-01]. Dostupné z: https://openaccess.thecvf.com/content_iccv_2015/papers/Girshick_Fast_R-CNN_ICCV_2015_paper.pdf

[23] REM, Shaoqing, Kaiming HE, Ross GIRSHICK, and Jian SUN. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. [online]. [cit. 2021-04-01]. Dostupné z: https://arxiv.org/pdf/1506.01497.pdf

[24] DAI, Jifeng, Yi LI, Kaiming HE, Jian SUN. R-FCN. Object Detection via Region-based Fully Convolutional Networks [online]. [cit. 2021-04-02]. Dostupné z: https://arxiv.org/pdf/1605.06409.pdf

[25] LIN, Tsung-Yi, Piotr DOLLÁR, Ross GIRSHICK, Kaiming HE, Bharath HARIHARAN and Serge BELONGIE. Feature Pyramid Networks for Object Detection. *Facebook AI Research (FAIR), Cornell University and Cornell Tech* [online]. [cit. 2021-04-02]. Dostupné z: https://arxiv.org/pdf/1612.03144.pdf

[26] Histogram of Oriented Gradients. *Intel Developer Zone* [online]. [cit. 2021-03-30]. Dostupné z: https://software.intel.com/content/www/us/en/develop/documentation/ipp-dev-reference/top/volume-2-image-processing/computer-vision/feature-detection-functions/histogram-of-oriented-gradients-hog-descriptor.html?language=ru

[27] SELIMAN, Amr, Yu-Hsin CHEN, Joel EMER, Vivienne SZE. Towards Closing the Energy Gap Between HOG and CNN Features for Embedded Vision. *Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology* [online]. [cit. 2021-04-11]. Dostupné z: https://arxiv.org/pdf/1703.05853.pdf

[28] LIU, Wei, Dragomir ANGUELOV, Dumitru ERHAN, Christian SZEGEDY, Scott REED, Cheng-Yang FU, Alexander C. BERG. SSD: Single Shot MultiBox Detector. *UNC Chapel Hill Zoox Inc. Google Inc. University of Michigan, Ann-Arbor* [online]. [cit. 2021-03-31]. Dostupné z: https://arxiv.org/pdf/1512.02325.pdf

[29] Figure of Single-shot Multibox Detector (SSD). *ResearchGate* [online]. [cit. 2021-03-31]. Dostupné z: https://www.researchgate.net/figure/Networks-of-a-Single-shot-Multibox-Detector-SSD-and-b-Multiheaded-Attention-Net_fig1_332904233

[30] FU, Cheng-Yang, Wei LIU, Ananth RANGA, Ambrish TYAGI, Alexander C. BERG. DSSD : Deconvolutional Single Shot Detector (SSD). *UNC Chapel Hill, Amazon Inc.* [online]. [cit. 2021-04-03]. Dostupné z: https://arxiv.org/pdf/1701.06659.pdf

[31] REDMON, Joseph, Santosh DIVVALA, Ross GIRSHICK,Ali FARHADI. You Only Look Once: Unified, Real-Time Object Detection. *University of Washington, Allen Institute for AI, Facebook AI Research* [online]. [cit. 2021-04-02]. Dostupné z: https://arxiv.org/pdf/1506.02640v5.pdf

[32] YOLO: Real-Time Object Detection *pjreddie* [online]. [cit. 2021-04-02]. Dostupné z: https://pjreddie.com/darknet/yolo/

[33] AlexeyAB/darknet *GitHub* [online]. [cit. 2021-04-02]. Dostupné z: https://github.com/AlexeyAB/darknet

[34] ultralytics/yolov5 *GitHub* [online]. [cit. 2021-04-02]. Dostupné z: https://github.com/ultralytics/yolov5

[35] LIN, Tsung-Yi, Priya GOYAL, Ross GIRSHICK, Kaiming HE, Piotr DOLLÁR. Focal Loss for Dense Object Detection. *Facebook AI Research (FAIR)* [online]. [cit. 2021-04-03]. Dostupné z: https://arxiv.org/pdf/1708.02002.pdf

[36] HUI, Jonathan. Object detection: speed and accuracy comparison (Faster R-CNN, R-FCN, SSD, FPN, RetinaNet and YOLOv3). *medium* [online]. [cit. 2021-04-03]. Dostupné z: https://jonathan-hui.medium.com/object-detection-speed-and-accuracy-comparison-faster-r-cnn-r-fcn-ssd-and-yolo-5425656ae359

[37] SpeechRecognition. *The Python Package Index* [online]. [cit. 2021-04-22]. Dostupné z: https://pypi.org/project/SpeechRecognition/

[38] Speech to Text. *Google Cloud* [online]. [cit. 2021-04-04]. Dostupné z: https://cloud.google.com/speech-to-text

[39] Speech to Text. *Microsoft Azure* [online]. [cit. 2021-04-04]. Dostupné z: https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text/

[40] Automatic Speech Recognition. *Houndify* [online]. [cit. 2021-04-04]. Dostupné z: https://www.houndify.com/technology

[41] alphacep/vosk-api. *GitHub* [online]. [cit. 2021-04-04]. Dostupné z: https://github.com/alphacep/vosk-api

[42] Build Natural Language Experiences. *Wit* [online]. [cit. 2021-04-04]. Dostupné z: https://wit.ai/

[43] Watson Speech to Text. *IBM* [online]. [cit. 2021-04-04]. Dostupné z: https://www.ibm.com/cloud/watson-speech-to-text

[44] Releases of Ubuntu. *Ubuntu* [online]. [cit. 2021-04-15]. Dostupné z: https://releases.ubuntu.com/

[45] Linux/Ubuntu - RealSense SDK 2.0 Build Guide. *Intel RealSense* [online]. [cit. 2021-04-16]
Dostupné z: https://dev.intelrealsense.com/docs/compiling-librealsense-for-linux-ubuntu-guide?_ga=2.107319492.71133901.1618582785-1752693200.1612431550

[46] librealsense/wrappers/python. *GitHub* [online]. [cit. 2021-04-15]. Dostupné z: https://github.com/IntelRealSense/librealsense/tree/master/wrappers/python

[47] NVIDIA CUDA Installation Guide for Linux. *NVIDIA Developer* [online]. [cit. 2021-04-15]. Dostupné z: https://docs.nvidia.com/cuda/archive/10.2/cuda-installation-guide-linux/index.html

[48] NVIDIA cuDNN documentation. *NVIDIA Developer* [online]. [cit. 2021-04-15]. Dostupné z: https://docs.nvidia.com/deeplearning/cudnn/install-guide/index.html

[49] tesseract-ocr/tesseract. *GitHub* [online]. [cit. 2021-04-22]. Dostupné z: https://github.com/tesseract-ocr/tesseract

[50] How to install OpenCV 4.2.0 with CUDA 10.0 in Ubuntu distro 18.04. *GitHub Gist* [online]. [cit. 2021-04-22]. Dostupné z: https://gist.github.com/raulqf/f42c718a658cddc16f9df07ecc627be7

[51] Releases. *OpenCV* [online]. [cit. 2021-04-22]. Dostupné z: https://opencv.org/releases/

[52] opencv/opencv_contrib. *GitHub* [online]. [cit. 2021-04-22]. Dostupné z: https://github.com/opencv/opencv_contrib/tree/4.4.0

[53] WolframAlpha. *wolframalpha* [online]. [cit. 2021-05-08]. Dostupné z: https://www.wolframalpha.com/

[54] espeak-ng/espeak-ng. *GitHub* [online]. [cit. 2021-05-13]. Dostupné z: https://github.com/espeak-ng/espeak-ng

[55] pyttsx3. *The Python Package Index* [online]. [cit. 2021-05-13]. Dostupné z: https://pypi.org/project/pyttsx3/

[56] gTTS. *The Python Package Index* [online]. [cit. 2021-05-13]. Dostupné z: https://pypi.org/project/gTTS/

[57] Stepper Motor: Bipolar, 200 Steps/Rev, 42×38mm, 2.8V, 1.7 A/Phase. *Pololu, Robotics & Electronics* [online]. [cit. 2021-05-12]. Dostupné z: https://www.pololu.com/product/2267

[58] DRV8825 Stepper Motor Driver Carrier, High Current. *Pololu, Robotics & Electronics* [online]. [cit. 2021-05-12]. Dostupné z: https://www.pololu.com/product/2133

[59] ARDUINO MEGA 2560 REV3. *arduino* [online]. [cit. 2021-05-12]. Dostupné z: https://store.arduino.cc/arduino-mega-2560-rev3

Zoznam skratiek

API	Application Programming Interface – rozhranie pre programovanie aplikácií
ASR	Automatic Speech Recognition – automatické rozpoznávanie reči
CNC	Computer Numerical Control – číslicové riadenie počítačom
CNN	Convolutional Neural Network – konvolučné neurónové siete
CPU	Central Processing Unit – centrálna procesorová jednotka
DNN	Deep Neural Network – hĺbkové neurónové siete
DSSD	Deconvolutional Single Shot Detector – dekonvolučný detektor jedného záberu
FPN	Feature Pyramid Networks – funkcia pyramídových sietí
GPU	Graphical Processing Unit – grafická procesorová jednotka
HOG	Histogram of Oriented Gradients – histogram orientovaných gradientov
IRH	Interactive robotic heads – interaktívne robotické hlavy
LTS	Long Term Support – dlhodobá podpora
OCR	Optical Character Recognition – optické rozpoznávanie znakov
OpenCV	Open source Computer Vision library – voľne dostupná knižnica počítačového videnia
R-CNN	Region-based Convolutional Neural Network – regionálne založená konvolučná neurónová sieť
R-FCN	Region-based Fully Convolutional Network – regionálne založená plne konvolučná sieť
RoI	Region of Interest – oblasť záujmu
SVM	Support Vector Machine – podporujúci vektorový stroj
SPP-net	Spatial pyramid pooling network – sieť združovaných priestorových pyramíd
SSD	Single Shot Detector – detektor jedného záberu
STT	Speech To Text – reč na text
TTS	Text To Speech – text na reč
YOLO	You Only Look Once – pozrieš sa iba raz

Zoznam príloh

arduino Zložka obsahuje kód nahratý do Arduina a jednoduché testovacie skripty.

robohlava Zložka obsahuje textový súbor README.md, v ktorom sú zhrnuté potrebné inštalácie na spustenie jednotlivých funkcii a taktiež obsahuje všetky použité a naprogramované skripty v jazyku Python.